TechNotes

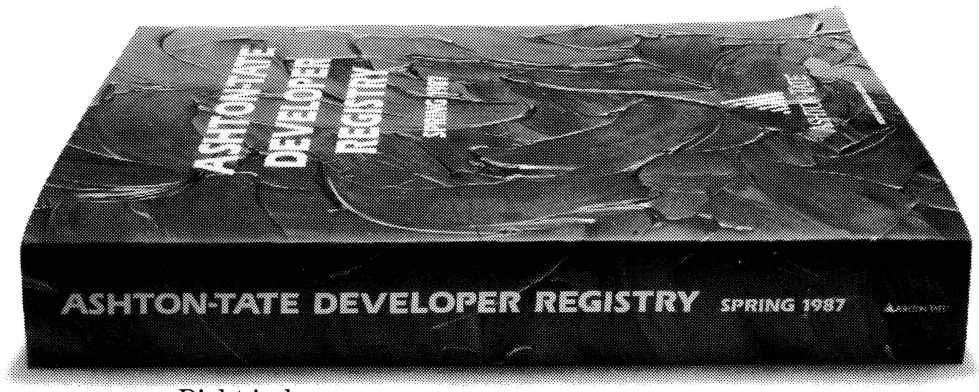
ASHTON-TATE'S TECHNICAL JOURNAL
FEATURING dBASE AND FRAMEWORK

AUGUST 1987
$6.00

- **Fine-tuning the IBM PC Network**
- **Memojoin**
- **Saving SET Environments**
- **Softstrips!**
- **Contest!**

TO BE LISTED IN THE NEXT REGISTRY, CALL TODAY! (213) 538-7358

If you need solutions, we're divulging sources.



Right in here.

It's a comprehensive listing of over 1,000 products and services from software developers and consultants nationwide. Cross-referenced and indexed so that you can easily find the specific solutions to meet your business software needs. And get the most out of Ashton-Tate® products like dBASE,® Framework,® RapidFile,™ and MultiMate.®

No matter what application you're looking for, you're likely to find it here. And to make sure you do, we've listed sources not only by business category, but also by product name, geographic locale, even area code. So you can't miss.

It's *the* sourcebook of Ashton-Tate software specialists, and you'll want to have it at your fingertips.

The *Developer Registry.* $24.95 at your local bookstore, or to order by phone call (800) 437-4329 Ext. 2208.

ASHTON·TATE®

In Colorado call (303) 799-4900, Ext. 2208. Trademarks/owner: dBASE, Framework, MultiMate, RapidFile, Ashton-Tate/Ashton-Tate Corporation. ©1987 Ashton-Tate Corporation. All rights reserved.

TechNotes

TABLE OF CONTENTS

AUGUST 1987

dBASE III PLUS	Fine-tuning the IBM PC Network Program *by Vincent Mendillo*	5
	Joining Databases That Include Memo Fields *by Michael B. Carlisle and Kenneth N. Getz*	18
	Generating SET Environment Programs *by Matt Johnson*	35
	dBASE III PLUS Tips	40
Framework II	Printing Row and Column Labels in a Spreadsheet *by Jeffrey McCrimon*	47
	Framework II Tips	52
RapidFile	Comparing dBASE III PLUS and RapidFile Functions *by Scott Haynes and James Bradbury*	54
	RapidFile Tips	60
Departments	Read Me First *by Brenda Johnson-Grau*	3
	Anomalies	62
	On-Line Services	66
	Letters	71
	Cumulative Index for January through August 1987	74

TechNotes

Publisher
Ashton-Tate Support Services
Jay Hanson, Director
Richard Goepel, Manager

Editor-in-Chief
Brenda Johnson-Grau

Editors
Kenneth N. Getz
James Bradbury

Technical Staff
Larry Abare, Richard Augelli, Stanley Ballenger, Rodney Barbati, Richard Bunter, Mike Carlisle, Fred Crownover, Michael Fiore, Rick Fillman, Peter Gold, Mina Gussin, Scott Haynes, Mike Hedblom, Clint Helton, Errol Jackson, Matt Johnson, Gail Kulikowsky, Steve Kurasch, Charles Litzell, Chris Lo Vasco, Jeffrey McCrimon, Jerry Mendes, Vincent Mendillo, Jeffrey Moore, Keith Mund, Joseph Perret, John Regen, Deborah Ridolfo, Cathy Shikatani, Steve Silverwood, Joseph Stolz, Naoma Stow, Philip Talsky, Mitra Toossi, Hiram Vega, Ralph Wakefield, Mark Ware, Quinn Wildman, Roger Wegehoft

Contributing Editors
Paul Bukauskas (International), Liz Burke (MultiMate), Steve Silverwood (On-Line Services)

Administrative Assistant
Kathy Smith

Production Staff
Gary Ashcavai, D. A. Gray Bare, Alecia C. Bernard, Tim Dunne, Darrell Evers, Renée Maynard, Patty Pesek, Ron Powell, Ann O. Taono

Product Development
Forefront Development Center: Marilyn McMahon, Greg Stikeleather
Glendale Development Center

DISCLAIMER NOTICE

The information contained in this publication is designed to help you develop and improve your programming skills and to assist you in using Ashton-Tate products. Ashton-Tate assumes no responsibility whatsoever for the uses made of any of the software code in this issue, whether modified or unmodified.

If you discover what you believe is an anomaly (as defined under **Anomalies**) not presented in this issue or others, please send a detailed description of the possible anomaly to:

Ashton-Tate Software Support Center
20101 Hamilton Ave.
Torrance, CA 90502-1319

If the reported anomaly can be reproduced by our technicians, it may be added to a future issue of ***TechNotes***.

The policies that are described in ***TechNotes*** apply only to the United States (and possessions), Canada, and Mexico. Residents of other foreign countries should contact a local Ashton-Tate representative in their country for policy information and technical support.

Ashton-Tate and the authors will not be liable for special, incidental, consequential, indirect or other similar damages, even if we have been advised of the possibility of such damages. This means we are not responsible for damages or costs incurred as a result of loss of time, loss of data, loss of profits or revenues, or loss of use of the software. In addition, we are not responsible for damages or costs incurred in connection with obtaining substitute software, claims by others, inconvenience or similar costs.

No warranties, either expressed or implied, are made regarding the contents of this publication, its merchantability or fitness for any particular purpose, all of which are hereby expressly disclaimed. Note, however, that some states do not allow the exclusion or limitation of warranties or of direct, incidental or consequential damages, so these warranty limitations may not apply to you.

In no event will our liability for any damages to you or any other person ever exceed the price paid for your subscription to ***TechNotes***, regardless of any form of the claim.

Additional statements by agents, employees, distributors and dealers of Ashton-Tate do not constitute warranties by Ashton-Tate and do not bind Ashton-Tate.

TechNotes (ISSN 0883-6493) is an Ashton-Tate Software Support Center publication containing technical reference information on Ashton-Tate and MultiMate products.

Entire contents Copyright © Ashton-Tate Corporation 1987. All rights reserved. Commercial reproduction of the material appearing in ***TechNotes*** is forbidden without prior written permission. Direct all requests for reproduction permission to Ashton-Tate Support Services, 20101 Hamilton Ave., Torrance, CA 90502-1319 ATTN: Brenda Johnson-Grau.

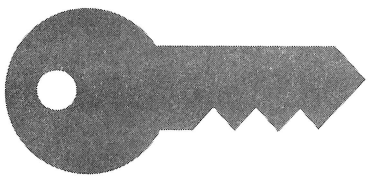
TechNotes is published monthly at a subscription price of $50.00 for one year. Additional rates: Canada and Mexico, $65.00; other countries, $75.00. Monthly issues are available in back issues for $10.00 each (1985 issues) and $7.00 each (1986 and 1987 issues). ***TechNotes*** is published by Ashton-Tate, 20101 Hamilton Ave., Torrance, CA 90502-1319. POSTMASTER: Send subscription inquiries and address changes to ***TechNotes*** Subscription Service, P.O. Box 3729, Escondido, CA 92025-0929.

Subscription Inquiries: Write: TechNotes Customer Service, PO Box 3729, Excondido, CA 92025-0929. Call: 800/828-2515 or 800/331-4164. From outside the US, call: 619/485-6535. When writing or calling, be sure that you have your current mailing label available. Your expiration date is listed on the far right of your mailing label.

dBASE, dBASE II, dBASE III, Ashton-Tate, and Framework are registered trademarks and RapidFile, dBASE III PLUS, Framework II RunTime, and FRED are trademarks of Ashton-Tate Corporation. Some of the trademarks of other companies mentioned in this issue appear for identification purposes only. MultiMate is a registered trademark and MultiMate Advantage, MultiMate GraphLink and the MultiMate Business Advantage Keyboard are trademarks of MultiMate International Corporation, an Ashton-Tate company. On-File is a trademark under license from Facts on File, Inc.

The Ashton-Tate Software Support Center's hours are 6:00 a.m. to 4:30 p.m., PST, Monday through Friday, except on Thursday when we close at 4:00 p.m. To reach our technicians for technical support, call (213) 329-0086. Be sure to have your software serial number ready for our operators.

READ ME FIRST



BY BRENDA JOHNSON-GRAU

Network aficionados—or at least those who use the IBM PC Network Program—will find much useful information in "Fine-Tuning the IBM PC Network," by Vincent Mendillo. It's a guide to getting the most speed and efficiency from that program when running it concurrently with the dBASE III PLUS ADMINISTRATOR.

JOINing two database files in dBASE is easily done—unless one or both contains a memo field. "Joining Databases That Include Memo Fields," by Michael B. Carlisle and Kenneth N. Getz, lets you have your JOIN and memo fields too.

dBASE programmers often need to make sure that their SET environments are properly maintained; trying to trace when STATUS, ESCAPE, EXACT, or HISTORY is SET OFF and ON can be exhausting. "Capturing and Restoring SET Environments," by Matt Johnson, provides two programs, one to capture the current SET environment in a database file and one to restore it once a subroutine has been executed.

Once in a while, you'll need to print the row and column labels of your Framework II spreadsheet. Perhaps you're having trouble keeping track of your cells or you need to create a map of your spreadsheet for a co-worker. As this type of printing isn't a built-in feature of Framework II, Jeffrey McCrimon has written a program to handle the feat: "Printing Row and Column Labels in a Spreadsheet."

This month's RapidFile article is actually a RapidFile/dBASE article. It's a reference guide for dBASE programmers who would like to better understand how to use RapidFile: "Comparing dBASE and RapidFile Functions" by Scott Haynes.

TechNotes On-Line

All the program code and many of the articles from each month's ***TechNotes*** can be downloaded from either The Source or CompuServe. To find the files on CompuServe, enter the correct data library (DL), 2 for dBASE III PLUS and 3 for Framework II. Issue a BROWSE command with the following syntax:

> BRO /KEY:AUG87 (Any ***TechNotes*** issue can be searched for in the same manner.)

Once the correct file is located, download it according to the protocol of the communications software you are using.

On The Source, you must first read the login banner to find out the date when the current month's articles were uploaded. By typing ATSIG from the command level, you will get the banner. Here is a portion of the current banner at the time of this printing:

A summary of the most recent uploads can be seen in Parti topic "ATSIG EXCHANGE" note 121. All files from the June 1987 issue of ***TechNotes*** may be searched on date for June 11.

The following dialog indicates how to search for June 11. User input is typed in boldface.

SEARCH ATSIG

Select by <K>eyword, <D>ate, <U>ser id, <FI>le name or <ALL>: **D**

Enter date: **June 11**

5 item(s) found for DATE = 11 Jun 1987

<N>arrow <E>xpand <REC>ap <R>ead <SC>an <C>ancel: **SC**

The target date for uploading of the information for each month's is the first of the cover month. Owing to production delays and the vagaries of our personal lives, this goal cannot always be met. We assure you, however, that the files will be uploaded soon thereafter.

Classifieds

For information on Classified advertising in ***TechNotes***, call 213/538-7517

Rates: $1 per word. $50 minimum. Display Classified are $60 per column inch.

Payment: Full payment must accompany each order. Mail copy and payment to:

TechNotes Classifieds
Attn: Brenda Johnson-Grau
20101 Hamilton Avenue
Torrance, CA 90502-1319

Agency Commission: No agency commission on regular or display classifieds.

Closing Date: Ads must be received 15th of the month two months prior to cover date of magazine (that is, August 15th is the deadline for the October issue).

Frequency discount: 10% discount given for four-month insertions.

Acceptance: Publisher reserves the right to accept or reject advertisements for Classifieds at their discretion.

Tear Sheets: A tear sheet showing classified ad as printed is sent without charge to each classified advertiser.

Phone Orders: Not accepted

User Groups: Ads are free. Name and address only. Limited to available space.

Fine-Tuning the IBM PC Network Program

BY VINCENT MENDILLO

The *performance* of a local area network is defined as the number of tasks that the network can perform during a specified time. To optimize the performance of your network, you need to do the same thing you would if you wanted to optimize the performance of a grand piano or a sports car—tune it up. In this article, we'll give you some tips on how to evaluate and improve the performance of the IBM PC Network Program. And we'll give you some tips on using the IBM PC Network with dBASE III PLUS.

PC Network Program performance is affected by many factors, which can be grouped into three categories:

- *Hardware* factors include the internal processor speed of the user and server PCs, the speed of devices such as disk drives and printers, the memory size, and the speed of the network adapter and cabling.
- *Volume* factors include the number of PCs on a network, the frequency of user-server requests, and the quantity and length of requests per time interval (the amount of network usage and the likelihood of collisions on the line).
- *Software* factors include the speed of the PC Network Program, of DOS 3.x, and of the applications installed on the network. Speed is often a function of the size and number of buffers.

In this article, we'll only look at how software factors affect network performance and how each element should be considered during your network tune-up.

DOS 3.1 and Config.SYS

The speed of DOS 3.1 and the PC Network Program can be tuned to specific user requirements. This does not mean that *any* requirement can be met. It does mean that carefully setting parameters can significantly improve performance.

In DOS, the only items that affect the network's performance are the entries in the CONFIG.SYS file (BUFFERS, FCBS, and FILES). CONFIG.SYS is a special DOS file that tailors DOS for using the PC's memory. If you already have a CONFIG.SYS file and install the PC Network Program program with the network installation aid diskette, CONFIG.SYS is changed to include the following:

BUFFERS = *xx*

where *xx* tells DOS how big to make the buffer pool in memory that it uses for many of its disk accesses. Increasing this value on nonserver computers won't improve network performance because the number of disk accesses that non-server machines make doesn't change when they are attached to the network.

On server machines, however, BUFFERS = is probably the most important performance-related parameter that you can specify in the CONFIG.SYS file. Depending on factors such as how much use the server's disk gets, what applications are installed, and how much memory the computer has, it's useful to test different values for this parameter. The default value is BUFFERS = 10. Try testing values of 15, 20, or 25. If the server is a PC AT, you should start testing with a value of at least 30.

Note: Before continuing on, be sure to review your IBM PC Network manual. See also "Configuring dBASE III PLUS" in the January 1987 issue of ***TechNotes.***

Terms and Concepts

Let's review some concepts and terms that relate to the PC Network Program's NET commands. First, let's review the different configurations that machines can have on the PC Network. The four configuration types for the IBM PC Network are REDIRECTOR (RDR), RECEIVER (RCV), MESSENGER (MSG), and SERVER (SRV), each of which requires a certain amount of memory and can perform specific types of tasks. Thus, the configuration that you select for your PC will depend on the size of the PC's memory, the availability of a fixed disk, and the tasks involved.

REDIRECTOR (RDR)

- can use network disks, directories, and printers (NET USE)
- can send messages (NET SEND)

RECEIVER (RCV)

- can do all that an RDR can do
- can receive and save (log) messages

MESSENGER (MSG)

- can do all that an RCV can do
- can use the network request keys (**Ctrl-Alt-Break**)

- can receive messages for other additional names
- can transfer (FORWARD) messages to other computers

SERVER (SRV)

- can do all that an MSG can do
- can share disks, directories, and printers

Menus and Commands

The IBM PC Network Program program provides two end-user interfaces. One is the system of NET menus; the other is the set of NET commands. Let's examine one of the NET commands: NET START.

NET START

NET START has 18 parameters. In order to set these parameters appropriately for the needs of your network, you need to understand what the parameters do and how their settings affect network performance. This knowledge, together with the discussion of actual parameter settings that follows it, will enable you to fine-tune a given network for better performance.

Command Format and Parameter Overview

You initiate all four configurations—SERVER, MESSENGER, RECEIVER, and REDIRECTOR—with the NET START command. You can use the menu interface or the fast-path method, or you can issue the command from the DOS prompt.

If the same PC Network Program setup is used frequently, it might be convenient to save it. You can save the setup from the menu interface and the program will automatically create an AUTOEXEC.BAT file with the settings you have selected.

With the NET START command, you can specify the value for from seven to eighteen parameters (seven for RDR/RCV configurations and eighteen for SRV/MSG configurations) that are used to tailor the system to your needs.

The syntax of the command is:

```
NET START configuration computername (/parameters)
```

where *configuration* can be SRV, MSG, RCV, or RDR; *computername* can be any name of up to 15 characters (no spaces or blanks); and (*/parameters*) are optional. The format of the parameters is:

```
/PAR:n(nnnn)
```

where "/" is the parameter separator, *PAR* is a three-character identifier, ":" is a value separator, and "*n(nnnn)*" is a value that is from one to five digits long (some parameters use only one-digit values, others use larger ones). You enter the command as follows:

```
NET START SRV ANYNAME /PAR:N/PAR:NNNN/PAR:Nk
```

To fine-tune your network, you might want to test different values for the 18 configuration parameters. However, be forewarned that this can be time-consuming, given that the effects of the changed parameters will be observable only by means of the NET START command—and that means a system reset. What's more, each time you restart a server you must restart all its remote users.

Given these caveats, the fastest way to change the settings of the parameters is to edit the AUTOEXEC.BAT file, which should contain the NET START command.

All of the eighteen NET START parameters are either organizational parameters or performance-related parameters.

A dBASE III PLUS Example:

Let's consider an IBM PC Network running concurrently with dBASE III PLUS Administrator. This example should help clarify how modifying certain parameters can improve performance results. We'll describe how both hardware and software factors apply to dBASE III PLUS on the network.

Our example system has an IBM AT as the file server, with two IBM XTs and one IBM AT as workstations. We are running a multiuser, dBASE III PLUS application that implements file and record locking. Let's make it an order-entry system where the XTs are adding orders with ten fields of information while simultaneously making various calculations. Meanwhile, the AT workstation will be making inquiries from the system with some light processing of reports. The initial configuration for the network settings is shown in Figure 1:

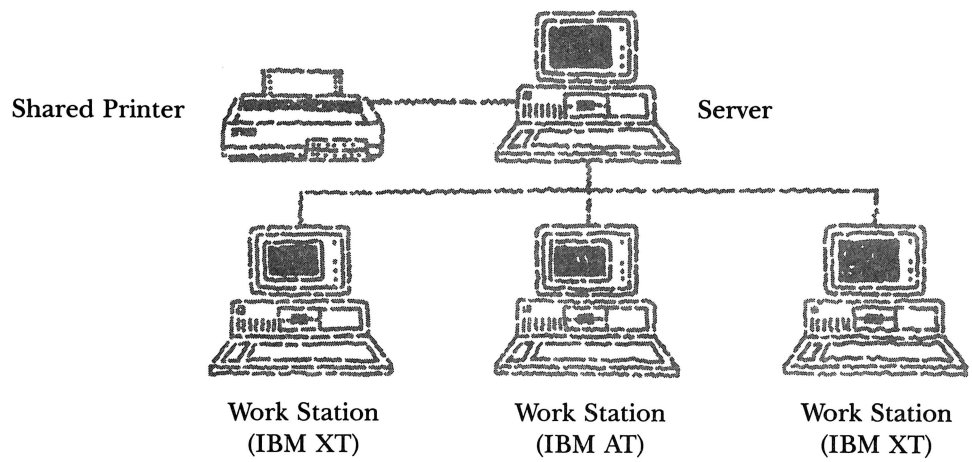


Figure 1.

Here are the intitial settings for CONFIG.SYS and AUTOEXEC.BAT for the server and for the workstations:

SERVER CONFIG.SYS	SERVER AUTOEXEC.BAT
Files = 45	NET START SRV < server >
Buffers = 30	NET SHARE DBASE = C:APPS\DBASE/RWC
FCBS = 16, 8	NET SHARE DBUSERS = C:\DBNETCTL.300/RWC

(Continued on page 17)

Organizational Parameters

Organizational parameters include all the parameters whose values depend on the organizational needs of the department where the network will be installed and/or whose values don't affect performance. These parameters have set values that probably can't or won't be changed for better performance. The following NET START parameters belong to this group:

/ASG /MBI /USN /SHB /SHL /SHR /RDR /SRV /SES /CMD

For example, the number of devices and directories might be determined by the number of files that are centrally administrated on a file server and that you have to look at or change from time to time—which leaves you little leeway.

When your server is not being used for local processing (that is, when it is a dedicated server), /TSI and /PRP can be considered organizational parameters as well.

Note: The dBASE ADMINISTRATOR only supports dedicated file servers.

Performance-Related Parameters

Performance-related parameters are those whose values directly or indirectly affect performance. These are the values that can be changed to find out the optimum setting for each machine on the network. The following parameters are performance-related:

/SES /CMD /NBC /NBS /PBX /PRP /TSI /PRP /USN /RQB /REQ /RDR /SRV

All but two (/TSI and /PRP) of these parameters define the different buffers used on the network.

Organizational versus Performance-Related Parameters

Some of the parameters, such as /SES, /CMD, /USN, /RDR, and /SRV, relate to both organizational and performance needs. In such cases, you must decide which consideration is more important and by what factor. If, for instance, performance is critical, then the parameter in question should be set to the minimum value that will still meet your organizational needs.

Memory Required

Most of the parameters will require additional memory if their value is increased. We assume that memory is not a limitation, and therefore we will not discuss how to set the values of the NET START parameters based on available memory.

Parameters

/SRV:*n* n = 1 to max 31
Maximum number of different computers that can be used (with NET USE) at one time
Default = 2 (3 on servers)
Configurations: All
Note: The maximum depends on configuration and other parameters (see /SES).

This parameter is primarily an organizational one. Its setting affects performance indirectly because it increases the number of NETBIOS sessions set with the /SES parameter, which itself affects performance directly. However, tests from IBM indicate that this parameter may not be of great importance. IBM suggests in the documentation that, at least at first, it be left at the default value. If this parameter proves to be important in a particular environment, remember the trade-off between organizational needs and network performance when you select its value.

The maximum value of 31 can be specified only on a REDIRECTOR configuration, which provides only basic functions. All the other configurations need to have more sessions reserved because of additional network functions. For these configurations, then, the maximum value for /SRV is decreased by the number of sessions that these additional functions require.

If you start a server and you specify /SRV:1, you won't be able to use a remote server. The redirector program module on the server machine will use /SRV:1 for establishing a session with the "local" server.

Note: It's important to distinguish between the *configuration* called REDIRECTOR and the *program module* called redirector. All configurations include the redirector module. The REDIRECTOR configuration performs the network tasks provided by the redirector program module.

/ASG:*n* n = 1 to 32
Maximum number of different network devices or directories that can be used (with NET USE) at one time
Default = 5
Configurations: All

This parameter is purely organizational. Its setting doesn't affect performance. It limits the user to a maximum of 32 network devices/directories at once. This is a comfortable limit. If the limit is reached by another user, it's possible to switch device/directories by stopping one and using another (NET USE command). Thus, the limit of 32 network device/directories is not an absolute one.

Note: The /SES and /CMD parameters are organizational variables, but they can affect performance. They influence the packet size used to transfer network requests and thus affect overall network performance. For light to moderate workloads these parameters might not be of great importance, and leaving them at their default values is recommended. Packet size may be more important in a heavily loaded network, however, or in a network where the principal activity is single-record access, such as a shared database application.

If these parameters turn out to be important in your particular environment, then remember the trade-off between organizational and network performance when you select their values.

/SES:*n* *n* = 1 to 32
Maximum number of NETBIOS sessions allowed on the computer at one time
Default = depends on configuration as follows:
16 on redirector and receiver
16 or (/USN + /SRV + 2) if bigger, on messenger
16 or (/RDR + /USN + /SRV + 2) on server

Note: Minimum settings for the parameter:
/SRV + 1 for redirector (2)
/SRV + 2 for receiver (3)
/SRV + /USN + 2 for messenger (3, /USN may be 0)
/SRV + /RDR + /USN + 2 for server (4, /USN may be 0)

The number of NETBIOS sessions specified causes one session entry for each session specified on the adapter RAM and an assignment of data buffers. The larger the number of sessions specified, the smaller the data buffers for the sessions. This reduces the packet size used to transmit requests over the line. As soon as the average network request exceeds the packet size, this request must be split into more than one packet. Since each packet includes some control information, this means that more bytes will be transmitted. Also, the likelihood of collision on the line per request will rise, and overall network performance will be lowered.

The assignment of the buffers is static and takes place as soon as the network is started. The /SES parameter value defines the potential number of sessions that can be established over the network rather than the actual number. You should therefore plan to meet your exact needs and avoid values larger than required for this parameter.

A session takes place between two names on the network that normally are located on different adapter cards. So, the maximum packet size on one card might be different from the packet size on the other. Network "handshaking" protocol ensures that the packet size used during the session will be tailored to the smaller of the two packets.

In most cases, for print and file requests, a server will determine the packet size used for a session because the server's /SES values will be higher than those of a remote user not configured as a server. For messages using session support (not broadcast datagrams), it might not be a server that determines the packet size because such messages can be sent and received by any user higher than a redirector.

REDIRECTOR (RDR): An RDR always has one active session for network maintenance. Beyond that, only the number of computers the RDR wants to use with the NET USE command will determine the number of sessions to be specified by /SES. Hence, the default of 16 for a redirector might be too high for your requirements. If so, adjusting /SES to your needs could improve performance.

RECEIVER (RCV): An RCV always has two active sessions: one for internal PC Network purposes and the other for "listening" for messages. In addition, only /SRV increases /SES. The default of 16 here is also potentially high, and you should adjust the values of /SES to your needs.

MESSENGER (MSG): Like the RCV, an MSG always has two active sessions. In addition, /SRV and /USN determine the value of /SES. The /SES parameter's default value of 16 might still be too high using this configuration. If so, adjust it to your needs. On this configuration, you don't have to worry about the setting of /SES if the default is too small; the adjustment of the parameter setting is handled automatically by the program itself.

SERVER (SRV): An SRV always has the same two active sessions as the RCV and MSG. Additionally, /RDR, /USN, and /SRV determine the value of /SES on this configuration. In this case, it is possible that the default value of 16 will be too small. This is because each active RDR using the SRV's resources requires one session to communicate with the server. The /SRV parameter setting is also increased if the local server's user has access to other server computers. Unless otherwise specified, the setting of the parameter /SES is done automatically by the system if its value increases to more than 16.

/CMD:*n* $n = 1$ to 32
Maximum number of NETBIOS commands that can be outstanding at one time
Default = depends on configuration
8 on redirector and receiver
8 or (/USN + 3) on messenger
8 or (/REQ + /USN + 4) on server

Note: Minimum settings of the parameter:
1 for a redirector
3 for a receiver
/USN + 3 for a messenger (3, /USN may be 0)
/REQ + /USN + 4 for a server (4, /USN may be 0)

This parameter allocates a fixed-length space on the adapter's random access memory for command entries. (Memory is also used from an NCB for each command.) The parameter specifies the number of outstanding NETBIOS commands that wait to be executed while another NETBIOS command executes.

On a REDIRECTOR, the /CMD parameter is always configured to at least one because there is a reserved entry for the NETBIOS command "send" used for the session with the server (/SES minimum = 1).

On a RECEIVER, the /CMD parameter is configured to at least three because there is a reserved entry for a NETBIOS command "send," one for the command "listen" to messages for a specific name, and one for the command "receive broadcast datagram."

On a MESSENGER, the same minimum number is reserved as in the RECEIVER configuration.

On a SERVER, the same minimum number is reserved as in the RECEIVER and MESSENGER configurations, plus at least one entry for the NETBIOS command "receiver" for receiving requests from a redirector.

The adapter card provides an open interface that can also be used by other applications besides the PC Network Program. NETBIOS commands can therefore be issued either by the PC Network Program or by other applications. Applications of this sort, according to the *IBM PC Network Program User's Guide*, are "directly accessing The Network."

During the NET START process, the network program resets the adapter and initializes it to hold a certain number of buffers for the sessions and commands, as specified by the /SES and /CMD parameters of the NET START command. Therefore, if you run applications that directly access the network, you should specify /SES and /CMD with values that cover the needs of your network configuration and the additional applications. Otherwise, the specified value of /SES would not reflect the real use of the adapter's memory. If this is the case, the program and/or the applications may fail when run together owing to insufficient resources on the adapter.

One example of an application that directly accesses the network is the PC NETWORK 3270 Emulation Program.

The /NBC and /NBS parameters are performance variables that can be used for fine-tuning. They set the number (/NBC = network buffer count) and size (/NBS = network buffer size) of the redirector's buffer for file transfer (read, write, open, close) from and to a server. /NBS, especially, can have a significant effect and can be recommended to users who want to improve network performance.

/NBC:*n* *n* = 1 to maximum 64
Maximum number of network buffers
the redirector module can use for file transfer from/to the server
Default = 3 (2 for computers with 128k or less memory)
Configurations: All, effective, or redirector

Check the number of files that can be opened concurrently by the application mix used and then set /NBC to the value that satisfies the needs of the mix (at least to the number of files that can be concurrently opened and accessed). Making this a large value will leave you a safety margin. Because increasing /NBC decreases /NBS (/NBC * /NBS < = 32K), you may need to select a setting that optimizes both parameter settings instead of maximizing each one.

For this parameter, both directions of data flow, from and to the server, must be taken into account.

Remember that a function such as a DOS COPY done on a server's disk opens two files. Some programs, such as dBASE III PLUS, might have up to 30 files open at one time. Other applications, when started, will open files and leave them open until the system is rebooted or the program itself is stopped.

/NBS:*n* *n* = 512 to 32K (bytes) (See Note.)
Maximum size of network buffer used by redirector (code) for file transfer from/to a server
Default = 512 (bytes)
Configurations: All, effective, or redirector

Note: The maximum depends on the setting of /NBC as follows:
/NBS * /NBC < = 32K
/NBS - /RQB bytes are not used.

/NBS defines the space to be allocated on the redirector for holding data before it is submitted to the adapter and then to the server or for holding data before it is given to the application coming from the adapter (from the server). The memory required by /NBS is treated as "cache memory" in the same manner DOS treats its buffers. The larger the size of the buffer, the smaller the time needed to read (write) from (to) slow devices (disks) in terms of the number of accesses.

In general, the larger the /NBS value the faster the network. A good initial setting is 3K. You can then modify that value if you wish to fine-tune this parameter.

There are also specific considerations that might influence the setting of /NBS. When the application requests I/O greater than /NBS size, it goes directly to the application buffer, bypassing the network /NBS buffer. When the request is less than or equal to the /NBS buffer size, the I/O is performed through the network buffer and data is moved from there to the application buffer. This is inefficient for sequential files with a request size equal to the buffer size. Decreasing /NBS will increase efficiency.

/PB*x*:*nn* x = 1 to 3 n = 80 to 16K (bytes)
Size of print buffer used to send print data to a shared network printer
Default = 1024
Configurations: All

This parameter is relevant to performance in spooling application requests. It is not effective if you are printing a file using NET PRINT, since the NET PRINT command uses its own buffer whose size is set internally within the code.

The parameter is set in the remote side (redirector) and is therefore restricted by the limit set by the /RQB on the server's side. Giving /PB a value bigger than /RQB would be a waste of memory.

/NBS is not involved in the process of printing an application print request. It would be involved only if the file to be printed resided on the server. In that case, the application file that needed to be printed would first be read from the server into the user's machine and would later return to the server as a print request.

The response time that a remote user gets for an application print request is the time elapsed from entering the request until it is queued. The actual printing is performed independently, and its performance is determined by the PSPRINT (background print module) print-buffer (set in /PRB) and the print priority set in /PRP.

The application prints to the buffer defined by /PB as if it were a printer. The speed at which the application is able to print to that buffer is influenced by its size. The time needed to transfer the request to the server is affected by the setting of /PB and by the setting of /RQB, which also specifies the limit to the setting of /PB. It is also influenced by the setting of /SES and /CMD.

However complex and unquantifiable this might be, it turns out that PB is one of the most important parameters that can be set for spooling of application print requests. To get good performance for application print requests, therefore, consider setting /PB equal to the length of the average print request issued by that application. Then have a look at what the current value of /RQB is. If it is less than what you need, consider adjusting it to your requirements. If that's not feasible, you should reduce /PB to the value set in /RQB. You will improve performance of remote application print requests by adjusting /PB and /RQB either to the size of an average print request or by maximizing them if this is not possible.

/TSI:*fb* b = 0 to 9 *f* = 0 to 9

Defines the time slice intervals for foreground and background activities in timer ticks

f, b value	Timer ticks
0	1
1	2
2	3
3	5
4	7
5	11
6	15
7	21
8	28
9	36

1 timer tick = 1/18.2 seconds

Default = 54

Configurations: Messenger, Server

This parameter is primarily relevant to performance. Its setting adjusts the relative response time of remote requests versus the response time of local requests made on the server computer. Here are some examples of foreground jobs:

- The user at the server/messenger entering DOS commands
- The user at the server/messenger entering NETWORK commands
- The user at the server/messenger running other applications
- The user working with the full-screen menu interface of the PC Network Program (for instance, printing files from the print queue)

Here are some examples of background jobs:

- Receiving messages from remote users
- Logging messages from remote users
- Remote user accessing a shared disk for file transfer
- Remote users spooling files to be printed

Note: /TSI applies to time left over after printing is completed.

Some examples of /TSI settings:

00 (/TSI = OFF) Execution time switching is fixed. Background tasks have priority. All background tasks are completed before switching to foreground tasks. Completion time for foreground tasks is very slow if the network activity is heavy. You might choose this setting for a machine that is mainly used as a file server and infrequently used for processing local requests.

91 Optimizes performance for foreground tasks. Background tasks are significantly slower if the foreground activity is high. This value of /TSI might be chosen for a machine that is not often used as a server/messenger and is used frequently for local purposes. (**Note:** Nondedicated servers are not currently supported by dBASE III PLUS.) It also might be used for a machine that is seldom used for foreground processing but needs to have good response time.

54 (Default) Allocates a fair amount of time to both foreground and background tasks. Leave this default for machines that are used as messengers, for file servers, and for local functions.

26 Gives a good response for background tasks but lets you do a minimal amount of foreground work with reasonably good response time. This is good for machines that are used mainly as printer servers. For these machine, the foreground task must be given some time to perform print queue management from the full-screen menu interface. Such machines should be left in the menu mode to avoid invoking the menus first.

Larger numbers for *f* and *b* generally result in less total system time allocated for the performance of a task because there is a greater chance that a task will not be interrupted before it is completed. However, numbers too large for *f* and *b* result in erratic response times. For instance, if you are using the server and try to use a network disk (a foreground task), but you have allocated a large amount of time to a background task, the foreground task receives no response. When the time interval for the background task is exhausted, the foreground task runs and receives a good response. Because the time interval for the foreground and the background tasks is long, you will notice an irregular response.

If you are not using the time slices that are defined with /TSI (for instance, the foreground time slices are not used because you don't do any foreground tasks), control is passed from the program to the request that needs to be processed, with priority to the background.

IBM PC Network Program configuration can be greatly influenced by the factors we have discussed. Whether they are set in the DOS CONFIG.SYS file or determined by the NET START command's parameter list, these factors can influence your network's performance. With practice and through experimentation, you should be able to apply the information presented here to fine-tune your own PC Network Program and get the best performance possible.

(Continued from page 8)

WORKSTATION CONFIG.SYS	WORKSTATION AUTOEXEC.BAT
Files = 20	NET USE D:\\name\DBASE
Buffer = 15	NET USE E:\\name\DBUSERS

The XT workstations adding daily orders use several commands that require database manipulation. This activity, in turn, affects network traffic. At the file server, a partial outline of the code used at the file server would be:

```
SET EXCLUSIVE OFF
USE Database
? FLOCK()
APPEND BLANK
@ 1,1 GET var
REPLACE Field WITH var
UNLOCK
```

Commands such as APPEND BLANK and REPLACE seem to take longer to execute than other commands. LOCKing commands, however, execute rapidly and thus don't slow down the network as much. One parameter of NET START that could improve the performance of file-manipulation commands such as APPEND BLANK and REPLACE is /NBS. Remember, /NBS defines "cache memory," so it operates like a DOS buffer. This means that a command like REPLACE needs less time to write to a device (such as a disk). Generally, file-intensive commands operate faster if this parameter is set up with a larger value. Therefore, the /NBS and /NBC parameters would have these values:

```
NET START SERVER /NBS:4K /NBC:10 ...
```

While this is taking place on the XT workstations, the AT workstation is generating reports. The reports are from both shared database files (non-exclusive) and proprietary database files (exclusive). The code might look something like this:

```
SET EXCLUSIVE ON
USE Database
SET DEVICE TO \\<computer name>\\ <printer name> = <destination>
REPORT FORM <report name> TO PRINT
```

Output that is spooled to the printer places demands on network resource time. The parameter of NET START that would improve performance in this situation is /PB. As it turns out, /PB is one of the most important parameters for dBASE III PLUS print requests. This setting is also intertwined with other parameters, such as /SES and /PRP. Print performance will improve by either adjusting /PB to the size of an average print request or by maximizing it.

Here is an optimal NET START command setting for such a networking environment:

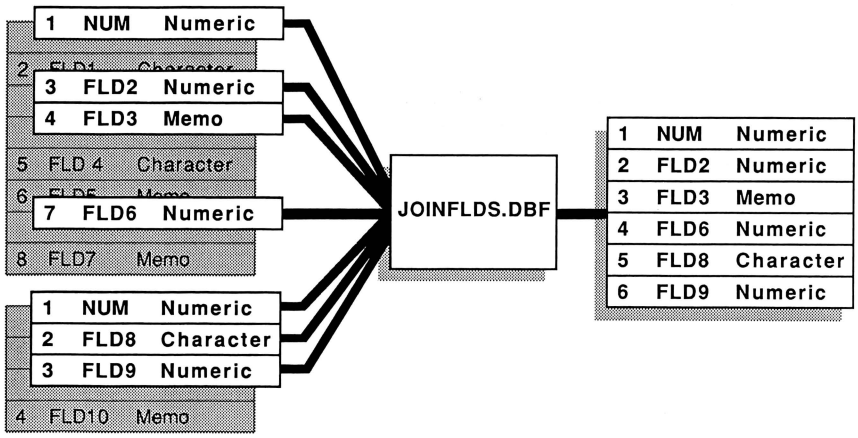
```
NET START SRV SUPPORT /PRP:1/SES:18/CMD:10/NBC:10/NBS:3036/TSI:18/USN:0/REQ:3/RQB:12
```

Joining Databases That Contain Memo Fields

BY MICHAEL B. CARLISLE AND KENNETH N. GETZ

dBASE III PLUS can easily JOIN two database files—unless either one contains a memo field. The programs presented here can accommodate the joining of database files that include memo fields. Memojoin.PRG, with its subsidiary files, Chkfiles.PRG, Makestru.PRG, Chgtype.PRG, and Mem2char.BIN, creates a combined database (.DBF) file that contains the same information that a normal dBASE III PLUS JOIN would have produced, as well as a combined memo (.DBT) file that contains all the memo information from the JOINed files.

JOIN can combine two databases on any logical condition. Owing to the constraints placed on us by the inclusion of memo fields, this is not possible with Memojoin. Memojoin can join two database files only if the key fields are equal.



Given the following database file structures:

Filea:

Field Name	Type	Width
NUM	Numeric	10
FLD1	Character	10
FLD2	Numeric	10
FLD3	Memo	10
FLD4	Character	10
FLD5	Memo	10
FLD6	Numeric	10
FLD7	Memo	10

Fileb:

Field Name	Type	Width
NUM	Numeric	10
FLD8	Character	10
FLD9	Numeric	10
FLD10	Memo	10

Performing the commands

```
SELECT 2
USE Fileb
SELECT 1
USE Filea
JOIN WITH Fileb TO Filec FOR Num = B->Num
```

will produce a file with the structure:

Filec:

Field Name	Type	Width
NUM	Numeric	10
FLD1	Character	10
FLD2	Numeric	10
FLD4	Character	10
FLD6	Numeric	10
FLD8	Character	10
FLD9	Numeric	10

Notice that all the memo fields have been lost in the JOIN process. Although the standard dBASE JOIN command gives you more control over the condition controlling the records to be JOINed (in this case, "Num = B->Num" could have been any valid dBASE Logical expression), you lose any memo fields.

Given the following file Joinflds.DBF:

Record#	DBF_NAME	FIELD_NAME
1	FILEA	FLD2
2	FILEA	FLD3
3	FILEA	FLD6
4	FILEB	FLD8
5	FILEB	FLD9
6	FILEB	NUM

the structure of the file formed by Memojoin would be:

Field	Field Name	Type	Width	Dec
1	NUM	Numeric	10	
2	FLD2	Numeric	10	
3	FLD3	Memo	10	
4	FLD6	Numeric	10	
5	FLD8	Character	10	
6	FLD9	Numeric	10	

with all the necessary memo fields intact. This method does preserve any memo fields, but it does not give you the same flexibility in choosing the JOIN condition as the dBASE JOIN.

Using Memojoin

To use Memojoin, you must first create the four command files necessary: Memojoin.PRG, Chkfiles.PRG, Makestru.PRG, and Chgtype.PRG. Use either MODIFY COMMAND or an ASCII text editor (one that introduces no formatting codes of its own). You also must create Mem2char.BIN, a machine-language module necessary for accessing the memo fields. Finally, you will need to create a list of the fields you plan to JOIN in a database named Joinflds.DBF. Once you create the command files, create the JOIN field list (Joinflds.DBF), and assemble Mem2char, you're ready to begin JOINing files.

Creating Joinflds

To tell Memojoin which fields from which database file you'd like to JOIN, create a database file named Joinflds.DBF to hold your list. The structure of the file should be:

Field	Field Name	Type	Width
1	Dbf_name	Character	8
2	Field_name	Character	10
** Total **			19

Once you've created this file, enter both the filename and the field name of each field that you want to include in the JOINed file. The filename should *not* include the .DBF extension.

Calling Memojoin

To call Memojoin, you need to pass four pieces of information as parameters:

First file to be JOINed
Second file to be JOINed
Output file
Key field

For example, suppose you want to JOIN File1 and File2, producing File3. Your criterion for the JOIN is that the field called Key be equal in both files. You would call Memojoin with the command line:

```
DO Memojoin WITH "File1", "File2", "File3", "Key"
```

What Memojoin Does

Memojoin first copies the two files to be JOINed. It then tests both files for the existence of the key field. If either file is missing the key field, Memojoin terminates and returns an error code.

After validating the files to be JOINed, Memojoin begins building the structure of the final result file, based on the fields list in Joinflds.DBF. Memojoin removes duplicate field names, tests for the existence of at least one memo field in the list, and makes sure that the defined field list will not exceed the dBASE III PLUS limit of 128 fields per record.

Memojoin then opens the temporary final result file and APPENDs the data from the files to be JOINed. This creates the combined .DBT file that holds the memo fields from both files (if there were memo fields in both files) and generates pointers that match the data in the combined .DBT file. Memojoin copies the combined .DBT file to a temporary file named Holding.DBT for safekeeping.

Memojoin then calls the machine-language routine, Mem2char, to modify the structure of the temporary result file, converting any memo fields to character fields so that records can be manipulated without losing the memo file pointers. The files to be JOINed are then USEd and ZAPped, and their memo fields are converted to character fields (again, using Mem2char). The data in the temporary result file corresponding to data from the first file is then APPENDed into the first data file (with the new memo pointers). This step is repeated for the second data file.

At this point Memojoin uses the JOIN command to join the database files, based on the user-supplied key field. This creates a database file with memo field pointers that correspond to the combined .DBT file formed earlier. Memojoin then uses Mem2char to convert the memo pointer fields back to the correct data type and then renames the necessary files.

Memojoin's final two steps are to COPY the JOINed file to the filename specified by the user as the final result file and to ERASE all temporary files.

Limitations

There are three limitations to Memojoin that must be considered. The first and most important is the amount of disk space required. Because of the number of temporary files Memojoin works with, you'll need four to five times the combined size of the two files to be JOINed in free disk space. A hard disk is absolutely necessary unless the databases to be JOINed are smaller than 60K combined.

Second, Memojoin performs its JOIN on an is-equal-to basis between common fields with the same name, data type, and size in the files to be JOINed. You *must* be sure that the two files do contain this key field before you try to JOIN them.

Third, owing to the amount of data manipulation involved, Memojoin takes considerable time to complete its job.

Error Handling

If a program error occurs while Memojoin is processing data, execution stops, all temporary files are erased, and an error number is returned in the PUBLIC memory variable called errornum. If a dBASE, DOS, or hardware error occurs (insufficient disk space, for example), the normal error mechanisms take over.

The error codes returned by Memojoin are:

Error number	Error condition
0	No error.
1	File to be JOINed does not exist.
2	Key field not found in file to be JOINed.
3	No memo fields to be JOINed. Use standard dBASE JOIN.
4	Fields list exceeds dBASE III PLUS limit of 128 fields.

If you call Memojoin from within your own application, make sure to check the value of *errornum* and have the application respond appropriately.

Setting up Mem2char

To set up Mem2char.BIN, create Mem2char.ASM using an ASCII text editor. Comments (anything following a semicolon in the program listing) are for human use only and will be disregarded by the assembler, so you needn't type them. All other punctuation and text is essential to the correct assembly of the .BIN file.

Once you have created the .ASM file, run the following series of programs to convert them into executable files. Though this code was written and tested exclusively with Microsoft's MASM 4.0, it should work correctly with version 2.0 or higher of IBM or Microsoft's MASM 2.0. (No versions lower than 2.0 of MASM and no versions of ASM.EXE will work because several changes have been made to the assembler.) You'll also need LINK.EXE and EXE2BIN.EXE (both supplied with DOS). Be sure that all of these programs are in the current directory or in your DOS path.

1. From the DOS prompt, type

   MASM Mem2char;

   to create an object file (Mem2char.OBJ).

2. Then type

   LINK Mem2char;

   to create Mem2char.EXE. Ignore the "Warning: no stack segment" message.

3. Type

   EXE2BIN Mem2char

   to convert the program into a format that dBASE III PLUS can LOAD.

4. Delete extraneous files:

   ERASE Mem2char.EXE
   ERASE Mem2char.OBJ

To CALL Mem2char, you must construct a memory variable that is a concatenation of the following information:

- "C" or "M"—the current field type
- The field number converted to a character (usually with the CHR() function)
- The full filename to act on (which can include a path)

For example, to convert the fourth field in the file Test.DBF from Memo to Character, use the following syntax:

```
memvar = "M" + CHR(4) + "Test.DBF"
CALL Mem2char WITH memvar
```

Remember that to see the change, you must USE the database file after this call.

If no error occurred, Mem2char returns a space character in the first character of memvar. If a DOS error occurs while Mem2char is trying to do its job, the character corresponding to the DOS error code will be returned as the first character of memvar. Mem2char also traps four typical errors not related to DOS. Looking at LEFT(memvar,1) should give one of the following values:

Error Code	Error Condition
CHR(32)	No error.
CHR(2)	File not found.
CHR(3)	Path not found or file doesn't exist.
CHR(4)	No file handle available.
CHR(5)	Access denied.
CHR(6)	Handle invalid or not open.
CHR(7)	The file was too small.
CHR(8)	The field number was out of range.
CHR(9)	The chosen field was not the correct type.
CHR(10)	The operator type was incorrect (Not C or M).
Any other character	Unspecified file error.

Although Mem2char does trap a number of common errors, it cannot handle a situation in which you try to convert a normal character field into a memo field. Attempting to do so will lead, at best, to unusual results.

```
* Program ...: Memojoin.PRG
* Author ....: Michael B. Carlisle
* Date .......: August 1, 1987
* Version ...: dBASE III PLUS
* Note(s) ...: Joins two database files with memo fields in either
*              or both files into a third file.
*
PARAMETERS file1, file2, outfile, joinfield
SET SAFETY OFF
LOAD Mem2char
PUBLIC errornum
errornum = 0
* ---Test user files.
DO Chkfiles WITH file1, "Flds1", joinfield
IF errornum = 0
   DO Chkfiles WITH file2, "Flds2", joinfield
ENDIF
IF errornum = 0
   * ---Build structure for result file.
   DO Makestru
ENDIF
IF errornum = 0
   * ---Make a combined .DBT file.
   CLOSE DATABASES
```

```
USE Temp3
APPEND FROM Temp1
cntr = RECCOUNT()
APPEND FROM Temp2
*
* ---Copy combined .DBT file to holding file.
CLOSE DATABASES
COPY FILE Temp3.DBT TO Holding.DBT
*
* ---Convert memo fields in Temp3 to character.
CLOSE DATABASES
DO Chgtype WITH "M", "Temp3.DBF", "Flds.DBF"
*
* ---Copy data for Temp1 to Temp4 to obtain new memo field pointers.
USE Temp3
COPY TO Temp4 FOR RECNO() <= cntr
*
* ---Convert memo fields in Temp1 to character.
CLOSE DATABASES
DO Chgtype WITH "M", "Temp1.DBF", "Flds1.DBF"
*
* ---Append Temp1 info with the new pointers.
USE Temp1
ZAP
APPEND FROM Temp4
*
* ---Copy data for Temp2 to Temp4 to obtain new memo field pointers.
USE Temp3
COPY TO Temp4 FOR RECNO() > Cntr
*
* ---Convert memo fields in Temp2 to character.
CLOSE DATABASES
DO Chgtype WITH "M", "Temp2.DBF", "Flds2.DBF"
*
* ---Append Temp2 info with the new pointers.
USE Temp2
ZAP
APPEND FROM Temp4
*
* ---Perform JOIN of Temp1 with Temp2.
CLOSE DATABASES
SELECT 1
USE Temp2
*
SELECT 2
USE Temp1
JOIN WITH Temp2 TO Temp4 FOR &joinfield = Temp2->&joinfield
CLOSE DATABASES
*
* ---Append JOINed data into Temp3.
USE Temp3
ZAP
APPEND FROM Temp4
*
* ---Re-establish .DBT file that matches new pointers.
CLOSE DATABASES
COPY FILE Holding.DBT TO Temp3.DBT
```

```
   *
   * ---Change memo fields in Temp3.dbf to memo type.
   CLOSE DATABASES
   DO Chgtype WITH "C", "Temp3.DBF", "Flds.DBF"
   *
   * ---Copy Temp3 to the specified result filename.
   USE Temp3
   COPY TO &outfile
ENDIF
*
CLOSE ALL
*
* ---Remove all temporary files.
ERASE Flds.DBF
ERASE Flds1.DBF
ERASE Flds2.DBF
ERASE Temp1.DBF
ERASE Temp2.DBF
ERASE Temp3.DBF
ERASE Temp4.DBF
ERASE Joinflds.NDX
ERASE Flds1.NDX
ERASE Holding.DBT
ERASE Temp1.DBT
ERASE Temp2.DBT
ERASE Temp3.DBT
ERASE Temp4.DBT
*
* ---Clean up.
RELEASE MODULE Mem2char
RETURN
* ---EOP Memojoin.PRG.
```

```
* Program ...: Chkfiles.PRG
* Author ....: Michael B. Carlisle
* Date ......: August 1, 1987
* Version ...: dBASE III PLUS
* Note(s) ...: Checks that the files to be joined exist and
*              ensures they contain the selected Key field. Creates
*              structure extended file with only the user-selected
*              fields as determined by the file Fld_List.DBF.
*
PARAMETERS filename, strufile, joinfield
*
* ---Test for existence of user file.
IF .NOT. FILE("&filename" + ".DBF")
   errornum = 1
   RETURN
ENDIF
*
* ---Make sure the key field is in the structure-extended file.
USE &filename
COPY TO &strufile STRUCTURE EXTENDED
```

```
USE &strufile
LOCATE FOR UPPER(Field_name) = UPPER(joinfield)
IF EOF()
   errornum = 2
   RETURN
ENDIF
*
CLOSE DATABASES
SELECT 2
USE Joinflds
INDEX ON UPPER(Field_name) TO Joinflds
SET FILTER TO Dbf_name = "&filename"
GO TOP
*
SELECT 1
USE &strufile
SET RELATION TO UPPER(Field_name) INTO Joinflds
GO TOP
*
* ---Remove fields from Structure Extended file that are not
* ---part of the user field list in the file Joinflds.DBF.
DO WHILE .NOT. EOF()
   SELECT 2
   IF EOF()
      SELECT 1
      DELETE
   ENDIF
   SELECT 1
   SKIP
ENDDO
CLOSE DATABASES
*
USE &strufile
PACK
CLOSE DATABASES
*
* ---Return to Memojoin.
RETURN
* ---EOP Chkfiles.PRG.
```

```
* Program ...: Makestru.PRG
* Author ....: Michael B. Carlisle
* Date ......: August 1, 1987
* Version ...: dBASE III PLUS
* Note(s) ...: Creates the structure of the new file, sets up memo fields,
*              and checks that the total number of fields does not exceed 128.
*              Duplicate field names are removed (except for the key field).
*
* --- Remove duplicate field names.
SELECT 2
USE Flds1
INDEX ON Field_name TO Flds1
*
SELECT 1
```

```
USE Flds2
SET RELATION TO Field_name INTO Flds1
GO TOP
DO WHILE .NOT. EOF()
   IF UPPER(Field_name) = UPPER(Flds1->Field_name) .AND.;
        UPPER(Field_name) <> UPPER(joinfield)
      DELETE
   ENDIF
   SKIP
ENDDO
CLOSE DATABASES
USE Flds2
PACK
COPY STRUCTURE TO Flds
CLOSE DATABASES
*
* ---Create working files.
CREATE Temp2 FROM Flds2
CREATE Temp1 FROM Flds1
*
* ---Append data from user files.
USE Temp1
APPEND FROM &file1
USE Temp2
APPEND FROM &file2
*
* ---Create structure for result file.
USE Flds
APPEND FROM Flds1
APPEND FROM Flds2
*
* ---Remove duplicate key field name.
LOCATE FOR UPPER(Field_name) = UPPER(joinfield)
CONTINUE
DELETE
PACK
*
* ---Test for existence of memo fields.
LOCATE FOR field_type = "M"
IF EOF()
   errornum = 3
   RETURN
ENDIF
*
* ---Test the number of fields.
COUNT TO many
IF many > 128
   errornum = 4
   RETURN
ENDIF
*
* ---Create temporary result file.
CLOSE ALL
CREATE Temp3 FROM Flds
*
* ---Return to Memojoin.
RETURN
```

```
* ---EOP Makestru.PRG
```

```
* Program ...: Chgtype.PRG
* Author ....: Kenneth N. Getz
* Date ......: 1 June 1987
* Version ...: dBASE III PLUS
* Note(s) ...: Procedure to traverse STRUCture EXTEnded file Flds.DBF, changing
*              field types to and from Character and Memo.
*              Called as DO Chgtype WITH <field type - 'C' or 'M'>,<filename>,
*              <structure extended file>
*
PARAMETERS mode, filename, strufile
USE &strufile
SET FILTER TO Field_type = "M"
GO TOP
DO WHILE .NOT. EOF()
   STORE mode + CHR(RECNO()) + filename TO x
   CALL Mem2char WITH x
   SKIP
ENDDO
RETURN
```

```
; Program ...: Mem2char.ASM
; Author ....: Kenneth N. Getz
; Date ......: August 1, 1987
; Version ...: dBASE III PLUS or Developer's Release
; Note(s) ...: This .BIN file, to be LOADed and CALLed from within dBASE,
;              allows you to change a field type from within a dBASE program.
;              To use it (once it's been LOADed), build a string using the
;              following information:
;
;                  1. C or M, the current field type.
;                  2. The field number to be changed.
;                  3. The full filename to be changed.
;
;              If no error occurs, the first character of the memory variable
;              passed in will be replaced with a space.  If there was an error
;              during the execution of Mem2char, the first character will be
;              replaced with one of the following codes:
;
;                 CHR(32): No error.
;                 CHR(2): File not found.
;                 CHR(3): Path not found or file doesn't exist.
;                 CHR(4): No file handle available.
;                 CHR(5): Access denied.
;                 CHR(6): Handle invalid or not open.
;                 CHR(7): The file was too small.
;                 CHR(8): The field number was out of range.
;                 CHR(9): The chosen field was not the correct type.
```

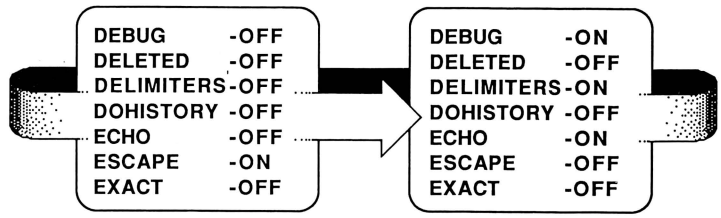
```
;               CHR(10): The operator type was incorrect.
;               Any other character: Unspecified file error.
;
;             To change field 3 in file Test.DBF from a memo field to a
;             character field, use the syntax:
;
;                 memvar = 'M' + CHR(3) + 'Test.DBF'
;                 CALL Mem2char WITH memvar
;                 IF LEFT(memvar,1) <> ' '
;                    DO CASE
;                    CASE LEFT(memvar,1) = CHR(7)
;                       * Handle this error
;                    CASE LEFT(memvar,1) = CHR(8)
;                    *
;                    * etc...
;                    OTHERWISE
;                       * Some other file error occurred.
;                    ENDCASE
;                 ENDDO
;
;             To see the changes made by Mem2char, you must USE the file
;             after you CALL Mem2char.
;*****************************************************************************
DOS     EQU     21h             ; Call to DOS.
CREATE  EQU     3Ch             ; Create a file.
OPEN    EQU     3Dh             ; Open a file.
CLOSE   EQU     3Eh             ; Close a file.
READ    EQU     3Fh             ; Read from a file.
WRITE   EQU     40h             ; Write to a file.
MOVE    EQU     42h             ; Move file pointer.
X_SIZE  EQU     7               ; Indicates the file was too small.
X_FNUM  EQU     8               ; Indicates field number out of range.
X_FTYPE EQU     9               ; Field not correct type.
X_OP    EQU     10              ; Wrong operator type.  (Not 'C' or 'M').
;-----------------------------------------------------------------------------
SAVE    MACRO   R1,R2,R3,R4,R5,R6,R7,R8
;
; This macro takes a parameter list and pushes each parameter register, in the
; order given, onto the stack.
;
;       USAGE:  SAVE AX,BX,CX...
;
        IRP     X,<R1,R2,R3,R4,R5,R6,R7,R8>
                IFNB    <X>
                        PUSH X
                ENDIF
        ENDM
        ENDM
;-----------------------------------------------------------------------------
RESTORE MACRO   R1,R2,R3,R4,R5,R6,R7,R8
;
; This macro takes a parameter list and pops each parameter register, in the
; opposite of the order given, from the stack.
;
;       USAGE:  RESTORE AX,BX,CX...
;
        IRP     X,<R8,R7,R6,R5,R4,R3,R2,R1>
```

```
                IFNB    <X>
                        POP X
                ENDIF
        ENDM
        ENDM
;-------------------------------------------------------------------------
GETFILE MACRO   MODE
LOCAL   L1,L2,L3
;
; This macro opens a file, putting the file handle into the variable HANDLE.
; If an error occurs, the error code is sent back to dBASE in the first
; character of the passed parameter.  The ASCII string for the file
; name is passed in by dBASE in the DS:DX pair.
;
;       USAGE:  GETFILE CREATE
;       USAGE:  GETFILE READ
;
        XOR     AX,AX                   ; Clear out AX.
        MOV     DX,OFFSET CS:FILENM     ; INT 21h needs address in DS:DX
        MOV     AH,MODE                 ; Get the correct mode in AH.
        MOV     AL,2
        CLC
        INT     DOS                     ; Call DOS.
        JC      L2                      ; Check for carry flag set (ERROR!)
L1:     MOV     CS:HANDLE,AX            ; If OK, move handle to storage.
        JMP     L3
L2:     MOV     CS:ERROR,AX             ; Otherwise, store error code.
L3:     NOP
        ENDM
;-------------------------------------------------------------------------
DOSFILE MACRO   @AH,@AL,@BX,@CX,@DX
;
; This macro simplifies calling DOS for all interrupt functions.
;
;       USAGE:  DOSFILE READ,0,HANDLE,1,<OFFSET CHECK>
;
        MOV     AH,@AH
        MOV     AL,@AL
        MOV     BX,@BX
        MOV     CX,@CX
        MOV     DX,@DX
        INT     DOS
        ENDM
;*************************************************************************
CODE    SEGMENT BYTE PUBLIC     'CODE'
;-------------------------------------------------------------------------
MEM     PROC    FAR
        ASSUME  CS:code
START:  JMP     ENTRY
;= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
        DB      13,'Mem2char.BIN',10,13
        DB      'Written by Kenneth N. Getz.',10,13
        DB      '(C) Copyright 1987, Ashton-Tate',10,13
        DB      26
OP      DB      ?
FLDNUM  DB      ?
FILENM  DB      65 DUP(0)               ; File name.
```

```
HANDLE  DW      0                       ; File handle.
ERROR   DW      0                       ; Error flag.
NOT_OP  DB      ?                       ; Target field type.
CHECK   DB      ?                       ; Temporary check storage.
POS     DW      ?                       ; Header position of current field.
HEADER  STRUC                           ; Structure for header information.
        VERSION DB      0
        DATE    DB      3 DUP(0)
        NUMRECS DD      0
        HEADSZ  DW      0
        RECLEN  DW      0
        RESRVED DB      20 DUP(0)
HEADER  ENDS
HEAD    HEADER  <>                        ; Initialization of structure.
;=======================================================
ENTRY:  SAVE    AX,BX,CX,DX,SI,DI       ; Save the environment.
        PUSH    DS
        MOV     HANDLE,0
        CALL    PARAMS                  ; Get the passed parameters.
        PUSH    CS                      ; Make DS = CS.
        POP     DS
        MOV     ERROR,32                ; Initialize error flag to space.
        CALL    GET_OP                  ; Find out what's changing to what.
        CMP     ERROR,32                ; See if error occurred.
        JNE     OUT                     ; If so, quit.
        GETFILE OPEN                    ; Open the necessary file.
        CMP     ERROR,32                ; See if error occurred.
        JNE     OUT                     ; If so, quit.
        CALL    GETHD                   ; Read in the file header.
        CMP     ERROR,32                ; See if error occurred.
        JNE     OUT                     ; If so, quit.
        CALL    MAKECHG                 ; Make the actual change.
OUT:    POP     DS                      ; Restore dBASE's DS register.
        CMP     CS:HANDLE,0             ; See if we ever opened a file.
        JZ      NO_CLS                  ; If not, don't close one.
        CALL    CLSFILE                 ; Otherwise, close the file.
NO_CLS: MOV     AL, BYTE PTR CS:ERROR   ; Send back the error byte.
        MOV     [BX],AL                ; Tell dBASE about error.
        RESTORE AX,BX,CX,DX,SI,DI       ; Restore the environment.
        RET                             ; And return to dBASE.
MEM     ENDP
;-------------------------------------------------------------------------
GET_OP  PROC    NEAR
;
; This procedure figures out what the target field type is, based on the
; current field type.
;
        MOV     NOT_OP,'M'              ; Assume we're changing C to M.
        CMP     OP,'C'                  ; See if that's true.
        JZ      OP_DONE                 ; If so, we're done.
        CMP     OP,'M'                  ; Otherwise, check M -> C.
        JNZ     NO_OP                   ; If not, we have an error.
        MOV     NOT_OP,'C'              ; Otherwise, target type is C.
        JMP     OP_DONE                 ; And we're done.
NO_OP:  MOV     ERROR,X_OP              ; Indicate error.
OP_DONE:
        RET
```

```
GET_OP  ENDP
;---------------------------------------------------------------------------
CLSFILE PROC    NEAR
;
; This procedure closes the file referred to by HANDLE.  If an error
; occurred while closing the file, the error code is passed back to
; dBASE in the first character of the calling parameter.
;
        SAVE    AX,BX
        MOV     BX,CS:HANDLE            ; Get the file handle.
        MOV     AH,CLOSE                ; Set up for Close.
        CLC                             ; Clear the Carry Flag.
        INT     DOS                     ; Call DOS.
        JNC     CLS_OK                  ; If Carry Flag still clear, we're OK.
        MOV     CS:ERROR,AX             ; Otherwise, send error back.
CLS_OK:
        RESTORE AX,BX
        RET
CLSFILE ENDP
;---------------------------------------------------------------------------
MAKECHG PROC    NEAR
;
; This procedure actually performs the work.  It reads the necessary byte
; from the file, changes it as necessary, and then writes it back out.
; If a field beyond the file's number of fields is specified, X_FNUM is
; returned.  If the chosen field's type does not match the current field
; type passed as a parameter, X_FTYPE is passed back.
;
        SAVE    AX,BX,CX,DX
        MOV     CL,5                    ; Set up for shifting.
        MOV     AL,FLDNUM               ; Get number of fields.
        XOR     AH,AH                   ; Clear out AH.
        SHL     AX,CL                   ; Multiply by 32.
        ADD     AX,11                   ; Add offset to get to field type.
        CMP     AX,HEAD.HEADSZ          ; See if we're past the header size.
        JBE     SIZE_OK                 ; If not, we're OK.
        MOV     ERROR,X_FNUM            ; Otherwise, send error back.
        JMP     NO_CHG
SIZE_OK:
        MOV     POS,AX                  ; Store away the file position.
        DOSFILE MOVE,0,HANDLE,0,POS     ; Go to that file offset.
        DOSFILE READ,0,HANDLE,1,<OFFSET CHECK> ; Read the byte.
        MOV     AL,OP                   ; Get the field type passed in.
        CMP     CHECK,AL                ; Compare it to what was in the file.
        JE      OP_OK                   ; If they're the same, we're OK.
        MOV     ERROR,X_FTYPE           ; Otherwise, indicate error
        JMP     NO_CHG                  ;       and jump out.
OP_OK:
        DOSFILE MOVE,1,HANDLE,-1,-1     ; Move back one byte.
        DOSFILE WRITE,0,HANDLE,1,<OFFSET NOT_OP> ; Write out the new byte.
NO_CHG:
        RESTORE AX,BX,CX,DX             ; Restore all registers.
        RET
MAKECHG ENDP
;---------------------------------------------------------------------------
GETHD   PROC    NEAR
;
```

```
; This procedure reads the file header into the structure HEAD.
; If an error occurs (32 bytes couldn't be read), ERROR will contain
; X_SIZE.
;
        SAVE    AX,BX,CX,DX             ; Read 32-byte file header.
        DOSFILE READ,0,HANDLE,32,<OFFSET HEAD>
        CMP     AX,CX                   ; See if we read all 32 bytes.
        JE      HD_OK                   ; If so, we're OK.
        MOV     ERROR,X_SIZE            ; Otherwise, we have an error.
HD_OK:  RESTORE AX,BX,CX,DX             ; Restore all registers.
        RET
GETHD   ENDP
;---------------------------------------------------------------------------
PARAMS  PROC    NEAR
;
; This procedure fills the data area with arguments passed on the command
; line from dBASE.  This routine is very dependent on the order of the
; variables in the data area.
;
        SAVE    AX,CX,DX,DS             ; Save the environment.
        MOV     CL,BYTE PTR [BX-1]      ; Get dBASE's param string length.
        XOR     CH,CH                   ; Clear the high byte.
        DEC     CX                      ; Don't want the last NULL byte.
        MOV     SI,BX                   ; Point SI to the string.
        MOV     DI,OFFSET OP            ; Point DI to the buffer area.
        REP     MOVSB                   ; Copy it all in to the code area.
        XOR     AL,AL                   ; Zero out AL.
        STOSB                           ; Make sure last byte is 0.
        AND     OP,0DFh                 ; Make sure OP is uppercase.
        RESTORE AX,CX,DX,DS             ; Restore the environment.
        RET
PARAMS  ENDP
;---------------------------------------------------------------------------
CODE    ENDS
        END     START
```



Generating SET Environment Programs

BY MATT JOHNSON

dBASE III PLUS programmers often need to be able to check the status of a specific SET variable before calling a routine that changes the status of that variable. For example, you may want to know whether CONFIRM is set on or off from within a program. Also, if a subprogram changes an environmental setting, you might want to be able to return this setting to its original value.

The program provided here, Getstat.PRG, writes the values of the current SET environment into a database file, Statfile.DBF. You can then query this database file to determine the value of various settings in your programs. Getstat can also write a program, Setparm.PRG, that resets your environment to its condition when Getstat was last run. It handles only variables that toggle between ON and OFF, skipping variables like DEVICE. Getstat is set up to handle only single-user environments, though it could easily be modified to handle multiuser environments as well.

Note: Getstat SETs TALK and CONSOLE OFF and ALTERNATE ON. It also deletes all the old records from Statfile.DBF and Temp.DBF.

Setup

To run the program, you must create two database files, Statfile.DBF and Temp.DBF, with the following structures:

Structure for database: C:Statfile.DBF

Field	Field Name	Type	Width	Dec
1	Setting	Character	9	
2	Toggle	Character	3	

Structure for database: C:Temp.DBF

Field	Field Name	Type	Width	Dec
1	Line	Character	80	

Statfile will contain each SET variable and its status. Temp is a scratch database file that reads the values of the status display.

Two utility programs, Getstat.PRG and Setparm.PRG, are presented along with a sample program, Sample.PRG, that demonstrates how you might use them in your programs. Sample simply stores the current status of the SET variables, lists out all the SET variables, and then turns on the status bar if it was on before the program was run. Be sure to have STATUS set on before you run Sample or the program won't be able to demonstrate its usefulness.

Getstat stores the listing of all SET variables in Statfile.DBF. Optionally, Getstat also creates another program, Setparm, to return the SET variables to the current settings when called. The syntax for using Getstat is:

DO Getstat WITH <logical value>

If the <logical value> is .T. (true), the program Setparm is created and stored for later use along with Statfile.DBF. If the <logical value> is .F. (false), the database file is filled, but the program is not written.

Chkstat.PRG is a small utility you can use once you have run Getstat. It uses the LOCATE command to find the correct keyword in the status listing, and returns the word ("ON" or "OFF") found in the corresponding Toggle field. If the keyword is not found, Chkstat returns a null string ("") which the calling program must handle.

```
* Program ...: Getstat.PRG
* Author ....: Matt Johnson
* Date ......: August 1, 1987
* Note(s) ...: Reads status info, writes settings to Statfile.DBF,
*              and optionally writes a program to reset the status values
*              after modification.
*
*              It skips over the following values which
*              are under the program's control:
*
*              SET ALTERNATE
*              SET DEVICE
*              SET CONSOLE
*
*              SAFETY is ON at the end of this program.
*
```

```
*             To use Getstat:
*
*                DO Getstat with <logical value>
*                   If <logical value> = .T., a .PRG file is written.
*                   If <logical value> = .F., only the .DBF file is written.
*
PARAMETERS makeprg                    && Makeprg controls the creation of
                                      && the program file.
SET TALK OFF                          && This will force TALK OFF in output.
SET ALTERNATE TO Statfile
SET ALTERNATE ON
SET CONSOLE OFF
LIST STATUS
SET CONSOLE ON
CLOSE ALTERNATE
SET SAFETY OFF
SELECT 1
USE Temp.DBF
ZAP
APPEND FROM Statfile.TXT SDF
*
SELECT 2
USE Statfile.DBF
ZAP
SET SAFETY ON
*
SELECT Temp
LOCATE FOR LEFT(Line, 9) = "ALTERNATE"
linenum = 1
DO WHILE linenum <= 8
   mline = Line
   position = 1
   itemnum = 1
   DO WHILE itemnum <= 4
      m_var = "m" + SUBSTR(mline, position, 4)
      &m_var = SUBSTR(mline, position + 13, 3)
      position = position + 19
      itemnum = itemnum + 1
      IF "m" <> TRIM(m_var) .AND. "mDEVI" <> m_var .AND.;
            "mALTE" <> m_var .AND. "mCONS" <> m_var
         SELECT Statfile
         APPEND BLANK
         REPLACE Setting WITH SUBSTR(m_var, 2, 9)
         REPLACE Toggle WITH &m_var
         SELECT Temp
      ENDIF
   ENDDO
   SKIP
   linenum = linenum + 1
ENDDO
IF makeprg
   SET ALTERNATE TO Setparm.PRG
   SET ALTERNATE ON
   SET CONSOLE OFF
   ? "* Program ...: Setparm.PRG"
   ? "* Date ......: " + DTOC(DATE()) + " " + TIME()
   SELECT Statfile
```

```
   GO TOP
   DO WHILE .NOT. EOF()
      ? "SET " + TRIM(Setting) + " " + Toggle
      SKIP
   ENDDO
   ? "RETURN"
   ? "* EOP Setparm.PRG"
   CLOSE ALTERNATE
ENDIF
CLOSE DATABASES
ERASE Statfile.TXT
SET CONSOLE ON
RETURN
* EOP Getstat.PRG.
```

```
* Program ...: Chkstat.PRG
* Author ....: Kenneth N. Getz
* Date ......: August 1, 1987
* Version ...: dBASE III PLUS, Developer's Release
* Note(s) ...: This program looks through Statfile.DBF for a specific
*              setting.  If found, it returns, through the variable value,
*              either ON or OFF.
*
*              This procedure assumes that no catalog is in use
*              because it uses work area 10.
*
*              To use:
*
*                 ret_val = ""
*                 DO Chkstat WITH "<setting>", ret_val
*
*                 <setting> is one of the settings in the LIST STATUS
*                 listing.  Ret_val holds the return value if a match
*                 was found.  If not, ret_val returns an empty string.
*
*
PARAMETERS paramstr, value
SELECT 10                                        && Assumes no catalog in use.
USE Statfile.DBF
LOCATE FOR Setting = UPPER(LEFT(paramstr,4))
value = IIF(EOF(), "", Toggle)
USE
RETURN
* ---EOP Chkstat.PRG.
```

```
* Program ...: Sample.PRG
* Author ....: Kenneth N. Getz
* Date ......: August 1, 1987
* Version ...: dBASE III PLUS, Developer's Release
```

```
* Note(s) ...: This is a sample program to show how you might use
*              Getstat.PRG and Chkstat.PRG from one of your own programs.
*
*
DO Getstat WITH .F.                     && No need for .PRG file.
SET STATUS OFF
USE Statfile.DBF
DISPLAY ALL
WAIT
USE
CLEAR
ret_val = ""                             && Initialize a return variable.
DO Chkstat WITH "STAT", ret_val         && Check to see if STATUS ON.
IF "" <> ret_val
   SET STATUS &ret_val
ENDIF
RETURN
* ---EOP Sample.PRG.
```

dBASE TIPS

dBASE III PLUS

Exclusive .OR.

dBASE's .OR. is nonexclusive. In other words, it returns true (.T.) in any case except when both conditions you are comparing are false (.F.).

Exclusive .OR. (normally called XOR) returns true (.T.) only if the two conditions you are comparing are different.

To XOR two conditions, you can use the following formula (where term_a and term_b represent two logical values):

```
? (term_a .AND. .NOT. term_b) .OR. (.NOT. term_a .AND. term_b)
```

The following code fragment prints out all the possible values for XOR:

```
x = "(term_a .AND. .NOT. term_b) .OR. (.NOT. term_a .AND. term_b)"
term_a = .T.
term_b = .T.
? ".T. XOR .T. = " + IIF(&x,".T.",".F.")
term_a = .T.
term_b = .F.
? ".T. XOR .F. = " + IIF(&x,".T.",".F.")
term_a = .F.
term_b = .T.
? ".F. XOR .T. = " + IIF(&x,".T.",".F.")
term_a = .F.
term_b = .F.
? ".F. XOR .F. = " + IIF(&x,".T.",".F.")
```

The printout shows:

```
.T. XOR .T. = .F.
.T. XOR .F. = .T.
.F. XOR .T. = .T.
.F. XOR .F. = .F.
```

Formatting Output

BY KEITH MUND

Columnar Listings

Here is a method you can use for listing a name and phone number file in the format:

```
Smith, Joe          phone number
Jones, Sally        phone number
```

The following code fragment demonstrates how to do this using the TRIM function.

(The value of the variable "col" determines the starting column of the phone number.)

```
USE clients
col = 20
SET HEADING OFF
LIST OFF LEFT(TRIM(Lastname) + ", " + Firstname + SPACE(col), col) + Phone
```

Using the Clients.DBF database file on the Sample Programs and Utilities disk, the printout of the first three records would look like:

```
Buckman, Claire     456-9059
Lisbonn, Rick       555-3344
Bicksby, Hank       966-1278
```

Numeric Formatting with "$"

The next two functions are designed to be used with REPORT FORM. If they were to be used in a program, some operations would be performed on a separate line to eliminate duplicate evaluation. In each case, "number" represents a numeric field in your file. The semicolons and carriage returns are included only for typographical clarity. Do not type them into your REPORT FORMs.

To place a single "$" character to the left side of a right-aligned number, place the following formula in the FIELD CONTENTS area of your report:

```
    STUFF(TRANSFORM(number, "9,999,999.99"),;
      LEN(TRANSFORM(number, "9,999,999.99")) -;
LEN(LTRIM(TRANSFORM(number, "9,999,999.99"))), 1, "$")
```

In a program, you could accomplish the same goal with the statements:

```
m = "TRANSFORM(number, '9,999,999.99')"
LIST STUFF(&m, LEN(&m) - LEN(TRIM(&m)), 1, "$")
```

Using this formula, the number -333.21 would be output as

```
$-333.21
```

while the number 333.21 would be output as

```
    $333.21
```

To place a single "$" character to the left side of a number with parentheses around negative numbers, place the following formula in the **CREATE REPORT COLUMN:CONTENTS** area of your report. Because this formula is so long and the length of the expression is limited to 254 characters, leave out any extra spaces as you type. If you run up against the 254-character limit, you might consider changing the variable names to include fewer characters. You could also abbreviate the dBASE function names by typing in only the first four characters of each.

```
IIF(number < 0,
    SPACE(13 - LEN(LTRIM(TRANSFORM(-number, "9,999,999.99")))) + "$(" +;
                LTRIM(TRANSFORM(-number, "9,999,999.99")) + ")",;
    SPACE(14 - LEN(LTRIM(TRANSFORM( number, "9,999,999.99")))) + "$" +;
                LTRIM(TRANSFORM( number, "9,999,999.99")))
```

In a program, you could accomplish the same goal with the statements:

```
m = "LTRIM(TRANSFORM(ABS(number),'9,999,999.99'))"
? IIF(num < 0, SPACE(13 - LEN(m)) + "$(" + m + ")", ;
              SPACE(14 - LEN(m)) + "$" + m)
```

Using this formula, the number -333.21 would be output as

```
    $(333.21)
```

while the number 333.21 would be output as

```
    $333.21
```

Search a Field for One String and Replace It with Another

Using the formula:

```
REPLACE ALL fieldname WITH ;
    STUFF(fieldname, AT(str1, fieldname), LEN(str1), str2);
    FOR str1 $ fieldname
```

you can replace any occurrence of str1 within the field "fieldname" with str2.

The following code fragment uses the first three records of Clients.DBF, a sample database file found on the dBASE III PLUS Sample Programs and Utilities Diskette.

```
USE Clients
str1 = "Rick"
str2 = "Sally"
field = "Firstname"
REPLACE ALL &field WITH STUFF(&field, AT(str1, &field), ;
       LEN(str1), str2) FOR str1 $ &field
str1 = "ck"
```

```
str2 = "x"
field = "Lastname"
REPLACE ALL &field WITH STUFF(&field, AT(str1, &field), ;
       LEN(str1), str2) FOR str1 $ &field
```

This is the database file as it originally appeared:

```
1 Claire   Buckman    8307 Santa Anita     Oxnard          456-9059
2 Rick     Lisbonn    1550 Keystone St.    Glendale        555-3344
3 Hank     Bicksby    4101 Peonia Rd       Flagstaff       966-1278
```

This is the resulting file:

```
1 Claire   Buxman     8307 Santa Anita     Oxnard          456-9059
2 Sally    Lisbonn    1550 Keystone St.    Glendale        555-3344
3 Hank     Bixsby     4101 Peonia Rd       Flagstaff       966-1278
```

APPEND FROM DELIMITED

APPEND FROM < Filename > DELIMITED WITH BLANK does not recognize blank fields and left-justifies the APPENDed data. When a blank field is APPENDed, the field is filled with data from the next field in the record. The data in the fields following the blank field are shifted into preceding fields. Leading blanks are removed from all APPENDed data. To illustrate,

```
.LIST
Record#  Field1   Field2   Field3
      1  March    March    March
      2  April             April
      3  May      May      May
      4  June     June     June

. COPY TO Test TYPE DELIMITED WITH BLANK
. ZAP
. APPEND FROM Test TYPE DELIMITED WITH BLANK
. LIST
Record#  Field1   Field2   Field3
      1  March    March    March
      2  April    April
      3  May      May      May
      4  June     June     June
```

In record 2, the data from Field3 shifted to Field2. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

CHR()

CHR(138) and CHR(141), "è" and "ì" respectively, produce unexpected results when displayed on the screen and cause .PRG files containing these characters to appear corrupted when edited with MODIFY COMMAND. Displaying CHR(138) causes the rest of the characters on that line to be suppressed. CHR(141) displays following characters on the next line. This occurs because dBASE III PLUS interprets CHR(138) and CHR(141) as a linefeed and a carriage return, respectively.

The dBASE III PLUS MODIFY COMMAND editor strips off the eighth data bit of all characters except CHR(141) when files are loaded. When CHR(138) is loaded into MODIFY COMMAND, it is transformed into CHR(10). Because CHR(10) is the MODIFY COMMAND end-of-line marker, data after the CHR(138) is not displayed. Executing a program file that formerly contained CHR(138) produces various errors when executed, depending on the command affected. When CHR(141) is loaded into the editor, characters following are displayed on the next line and the line status, on the right side of the screen, reveals that the line is continued. The CHR(141) remains in the file unless it is deleted or overwritten. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

dBCODE

dBCODE does not trap illegal characters in filenames when encoding command files. Executing a dBCODEd command file that references an illegal filename produces the error message "Unrecognized phrase/keyword in command." [36] or "File does not exist." [1].

DIRECTORY

DIRECTORY is an undocumented dBASE III PLUS command that performs identically to the dBASE DIR command.

INSTALL.BAT

The files required to run dBASE III PLUS version 1.1 are not documented. They are:

DBASE.EXE
DBASE.OVL
DBASEINL.OVL
DBASE.MSG
CONFIG.SYS
(In the root directory of the boot disk.)

Additional files needed by dBASE III PLUS (version 1.1) to function fully but not required to load are:

HELP.DBS
ASSIST.HLP
CONFIG.DB

SET CATALOG TO ?

Attempting to open a catalog file that has the same name as an open database file or assigned alias name produces the error message "ALIAS name already in use." [24]. To avoid the error message, assign a different alias to the database file or close the database file before executing SET CATALOG TO. For example,

```
USE Test
SET CATALOG TO Test
```

produces the error message. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

dBASE ADMINISTRATOR

Using the Novell MAP Command

The powerful Novell MAP command can assist you in customizing your network to take full advantage of dBASE III PLUS. The syntax for the MAP command is outlined below, as well as some examples of how to interface the command with dBASE.

Syntax:

MAP — Displays the current drive mappings for your workstation.

MAP *drive:* — Displays the current mappings for the specified drive.

MAP *drive:* = *directory* — Maps the specified directory to the specified drive letter. This command may also be nested: MAP *drive:* = *directory; drive:* = *directory*

MAP *directory* — Maps the specified directory to the default drive.

MAP *drive:* = — Maps the default directory to the specified drive.

MAP INSERT *search drive:* = *directory* — Inserts a new search drive that is mapped to the specified directory. It positions this search drive into the correct sequence in the search mappings.

MAP DEL *drive:* — Removes a drive mapping.

The MAP command is very similar to the DOS SUBST command. The purpose of the MAP command is to establish logical and search drives. A logical drive is a subdirectory that can be accessed by specifying the drive designation instead of the path. A search drive is similar to a logical drive except that any directory specified as a search drive is added to the path that DOS searches when executing a command that is not found in the current directory. A search drive is also assigned a logical drive designation. The maximum number of logical drives that can be MAPPED is 26. The maximum number of search drives is 16.

Note that NetWare security will not allow you to access any files in directories to which you have not been assigned rights. Therefore, mapping a drive to a directory that you do not have rights to is a waste of a drive designation.

When mapping search drives, the logical drive designations that are automatically assigned start at Z: and work backward in the alphabet. Therefore, when assigning logical drives, you should start at A: and work down.

dBASE Examples:

You have installed dBASE ADMINISTRATOR onto a file server in a subdirectory called SYS:DBA. When you installed, the DBNETCTL.300 subdirectory was created, as SYS:DBNETCTL.300. You also created a sub-directory called SYS:\DBA\APPS that contains all your applications and data files. If there were no other drive mappings, the following mappings would be suggested:

```
MAP F:=SYS:DBA
MAP G:=SYS:\DBA\APPS
MAP SEARCH1:=SYS:DBNETCTL.300
```

With these mappings, you could log into drive F: and start DBA. You could then use the SET DEFAULT TO G: command to change directories to your applications. Note also that it would not be necessary to use the #DF = option on the DBA command line, because the DBNETCTL.300 directory will be located by dBASE since its files are a part of a search drive. (You can abbreviate a search mapping by replacing SEARCH1 with S1.)

When you log out from a file server, the logical drive mappings are cleared, and your path is restored to the way it was before you logged in.

Files to Erase When Reinstalling the dBASE ADMINISTRATOR

The following is a list of files and directories that must be deleted before attempting to reinstall the dBASE Administrator:

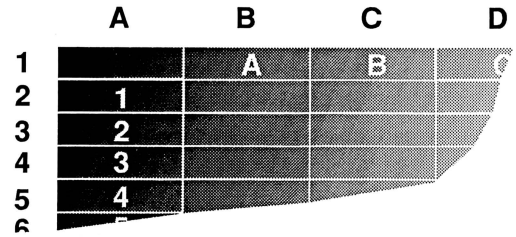
1. DBA.COM in the dBASE subdirectory.

2. The \DBNETCTL.300\XXXXXXXX.SUB\VDF0300.HUM file, where XXXXXXXX is variable.

3. All files in the \DBNETCTL.300 subdirectory.

4. The \DBNETCTL.300\XXXXXXXX.SUB subdirectory.

5. The \DBNETCTL.300 subdirectory.



Printing Row and Column Labels in a Spreadsheet

BY JEFFREY MCCRIMON

When Framework II prints a spreadsheet, it does not print the standard row-and-column labels (A, B, C, 1, 2, 3, and so on) that are part of the spreadsheet screen display. Figure 1 shows you how a spreadsheet appears on the screen, and Figure 2 shows you how the same spreadsheet appears when printed out.

```
          A          B          C          D          E        ...
1                 January    February    March      April      ...
2   Rent           500.00     500.00     500.00     500.00     ...
3   Auto           300.00     300.00     300.00     300.00     ...
.    .               .          .          .          .         .
.    .               .          .          .          .         .
```

Figure 1

```
          January  February March    April
Rent        500.00   500.00   500.00   500.00
Auto        300.00   300.00   300.00   300.00
```

Figure 2

Occasionally, you'll need to include the standard row-and-column labels in printouts of your spreadsheets. The macro presented here, AddLabel.FW2, allows you to print a spreadsheet with these row and column labels, as shown in Figure 3.

```
    A         B         C         D         E         F
1           January  February March    April
2 Rent        500.00   500.00   500.00   500.00
3 Auto        300.00   300.00   300.00   300.00
4
```

Figure 3

How AddLabel Works

AddLabel adds a new row and column to your spreadsheet, **MOVEs (F7)** the data from row 1 to row 2, and then **MOVEs (F7)** the data from column A to the new column (B). It correctly **SIZEs (F4)** both the new column and the old column, if necessary. AddLabel uses **@fill** along with a user-defined function, FillCol, to place the standard labels in the first row and column of your spreadsheet. **@fill** fills in the row labels and FillCol fills in the column labels. The user-defined function Adjust **SIZEs (F4)** the first two columns of the spreadsheet as necessary, to retain your original column width for column A.

Setting Up AddLabel

To set up AddLabel, create an empty frame in the Macros section of your Library, name it with an **Alt-** < key > combination, and enter the AddLabel program into the formula area of the frame. Then create an empty frame in the User-Defined functions section of your Library, name it FillCol, and enter its formula in the formula area of the frame. Create another empty frame in the User-Defined functions section of your Library for Adjust. Be sure to save the Library frame.

To execute AddLabel, highlight a spreadsheet frame and press the appropriate **Alt-** < key > combination.

Note: The keystroke-recording macro, {Alt-K}, presented in the July 1987 issue of ***TechNotes*** on page 42 would be very helpful when you are entering this program code. We suggest that you prepare {Alt-K} before you enter the program code for AddLabel.

```
; AddLabel.FW2
;
; by Jeffrey McCrimon
; August 1, 1987
;
; This macro adds standard row and column labels inside
; a spreadsheet frame so that they can be printed.
;
@local(start, end, form),
;
; Check for frame type and location.
@if(@frametype <> 6,
     @return(
          @list(
               @beep,
               @eraseprompt,
               @prompt("This is not a spreadsheet. Press any key to continue..."),
               @nextkey,
               @eraseprompt
          )
     )
),
; Check for spreadsheet name.
@if(@mid(@setselection,1,2) = "[]",
     @return(
          @list(
               @beep,
               @eraseprompt,
               @prompt("The spreadsheet must have a name. Press any key to continue..."),
               @nextkey,
               @eraseprompt
          )
     )
),
; Insert a row and column.
@performkeys("{Dnlevel}{Ctrl-Home}{Ctrl-C}R1{Return}"  &
             "{Uparrow}{F6}{End}{Return}{F7}{Dnarrow}" &
             "{Return}{Ctrl-Home}{Ctrl-C}C1{Return}"
```

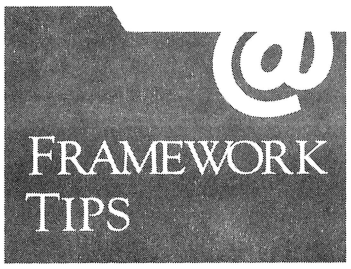
```
),
; Size the first two columns correctly.
@Adjust,
;
; Switch the first two columns.
@performkeys("{Leftarrow}{F6}{Ctrl-Pgdn}{Return}"      &
             "{F7}{Rightarrow}{Return}{Ctrl-Home}{Rightarrow}"
),
;
; Build formula to place traditional column labels in Row 1.
@eraseprompt,
@prompt("Creating column coordinates, please wait..."),
start := @setselection,
@performkeys("{End}"),
end   := @setselection,
form  := "@execute(" & end & ", @Fillcol(" & start & ":" & end & "))",
@form,
;
; Center the labels.
@performkeys("{Ctrl-Home}{F6}{End}{Return}{Ctrl-N}M{Ctrl-Home}{Dnarrow}"),
;
; Build formula to place the traditional number labels in Column 1.
@eraseprompt,
@prompt("Creating row coordinates, please wait..."),
start := @setselection,
@performkeys("{Ctrl-Pgdn}"),
end   := @setselection,
form  := "@execute(" & end & ", @fill(" & start & ":" & end & ",1,1))",
@form,
;
; Right-justify the labels.
@performkeys("{Ctrl-Home}{F6}{Ctrl-Pgdn}{Return}" & "{Ctrl-N}R"),
;
; Exit.
@performkeys("{Ctrl-Home}{Ctrl-Uplevel}"),
@eraseprompt,
@prompt("Finished.")
```

```
; Fillcol.FW2
;
; by Jeffrey McCrimon
; August 1, 1987
;
; This user-defined function places standard
; labels (A, B, C) in a specified region.
;
; Syntax: @Fillcol(region)
;
@local(char_counter, char, loop, count),
char_counter := 65,
loop := 0,
count := 0,
@while(@get(@item1) <> #NULL!,
```

```
    char := @chr(char_counter),
    ; Are we past the label 'Z'?
    @if(loop > 0,
        ; Yes, place label in the format 'AA'
        char := @chr(65 + count) & char
    ),
    @put(@item1,char),
    char_counter := char_counter + 1,
    ; Are we past the label 'Z'?
    @if(char_counter > 90,
        ; Yes, reset to the label 'A'.
        @list(
            char_counter := 65,
            loop := loop + 1,
            @if(loop > 1, count := count + 1)
        )
    ),
    @next(@item1),
)
```

```
; Adjust
;
; This macro expects the cursor to be on the right of the
; two columns to adjust.  It then swaps the widths of the two
; columns.  For example, the cursor is on column AC, the width of
; column AC is 10 and the width of column AB is 20.  When
; this function is done, the cursor will be on column AC, the
; width of column AB will be 10 and the width of column AC
; will be 20.
;
@local(width1, width2),
@performkeys("{F4}"),
width2 := @panel1,
@performkeys("{Return}{Leftarrow}{F4}"),
width1 := @panel1,
@if(width1 <> width2,
    @performkeys("{F4}{Ctrl-Home}" &
                 @rept("{Rightarrow}", width2 - 1) &
                 "{Return}{Rightarrow}{F4}{Ctrl-Home}" &
                 @rept("{Rightarrow}", width1 - 1) &
                 "{Return}"
    ),
    @performkeys("{Return}")
)
```

Limitation: As currently written, Fillcol will handle only spreadsheet columns up to column ZZ. To handle more columns would require both changing the loop structure of Fillcol and modifying the default number of **@while** repetitions through the Framework II Setup program.



@sense and Telecommunications

The Framework II **@sense** function is usually used to return a #TRUE or #FALSE value based on the status of the menu selection tested. You can also use it to indicate whether or not telecommunications is active. When you are on-line and connected, **@sense**(#ATSB) returns the name of the active Telecomm frame. If there is no active Telecomm frame, **@sense**(#ATSB) returns #N/A!.

Date Import

When you import a database file from dBASE III or dBASE III PLUS, date fields are imported in the same format as they are stored in dBASE: "YYYYMMDD". The following macro presents one possible way to convert dBASE dates into Framework II dates.

To set up the macro, create a frame in your Library Cabinet and assign it a name in the format {**Alt-** < keyname > }, where *keyname* is any alphabetic character. We have assigned {**Alt-C**}, but you can name it whatever you like. Type the following code into the formula area of the frame. When you're done, save your Library Cabinet by pressing {**Alt-F2**}.

After you've imported your database, position the cursor on the column heading of the field you'd like to convert. Press the Alt-key combination you assigned to the macro. The macro also **SIZEs (F4)** the column so the new date format will show completely in the column.

```
; {Alt-C}
;     Converts dBASE imported dates into Framework II format.
;    by Kenneth N. Getz
;    August 1, 1987
;
@local(string,date),
@prompt("Working.  Please wait..."),
@echo(#OFF),
@performkeys("{Dnarrow}"),
@isabend(#FALSE),
@while(@not(@isabend),
     @performkeys(" {F6}{End}{Backspace}"),
     string := @textselection,
     date := @mid(string, 1, 4) & "," & @mid(string, 5, 2) & "," &
             @mid(string, 7, 2),
     @performkeys("{Del}{Return}{F2}" &
                  "@date(" & date & "){Return}" &
                  "{Dnarrow}"),
),
```

```
@performkeys("{Ctrl-Pgup}{F4}{Rightarrow}{Rightarrow}{Return}"),
@echo(#ON)
```

@dbasefilter

The **@dbasefilter** function does not allow you to use a local variable as part of the filter formula. For example, in dBASE III PLUS, create a database file called Members.DBF with one field labeled Name and enter several names into the database file. Next, quit dBASE III PLUS, run Framework II, and select Members.DBF from the drive cabinet. Finally, change the filter formula on the Members frame to

```
@local(mname), mname := @inputline("Enter a name on which to filter."),
@dbasefilter("Members.DBF",Name = mname)
```

When this formula is recalculated, no records are appended from the dBASE III PLUS database file, regardless of the value entered in response to the **@inputline** function.

The **@dbasefilter** function causes the filter formula, "NAME = mname", to execute outside the database frame's formula. This way, the local variable is not available to the called formula (**@dbasefilter**); this results in the filter formula evaluating to #FALSE for every record.

The solutions are to make mname a reference instead of a local variable, or to create a user-defined function to filter the database. For example, on the border of the frame labeled Members, enter

```
@local(mname, user_def_func),
mname := @inputline("Enter a name on which to filter"),
user_def_func := "@dbasefilter(""" &
    "Members.DBF""" & ",Name = """ & mname & """)",
@user_def_func
```

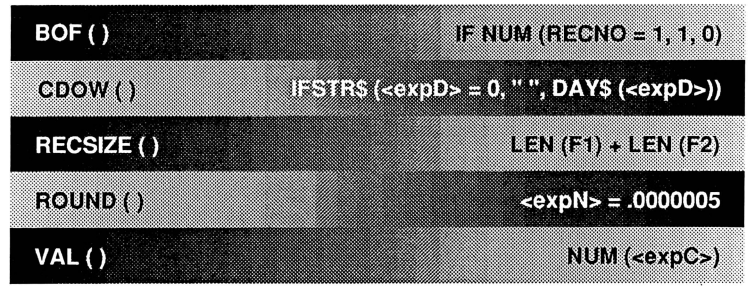
Note that a set of triple quotes indicates an embedded quotation mark.

Missing Operand Message

Because attributes can be turned off but not deleted, Framework II displays the error message "Missing Operand" when the first visible byte of a formula is attributed (bold, italics, and so on) and the entire formula is deleted.

Also, because the **Alt-F7** Cut macro writes the **Extend Select** begin and end status bytes to the Cut Paste Buffer, this same error message is displayed if the first byte of the formula is placed there by the **Alt-F8** Paste macro, and the entire formula is then deleted. Because these bytes are not visible, Framework II treats them in the same manner as attribute bytes.

To avoid this, edit the formula on the frame and turn the **Words:Normal** option ON before reentering a formula. This will clear all attribute bytes.



Comparing dBASE III PLUS and RapidFile Functions

BY SCOTT HAYNES AND JAMES BRADBURY

RapidFile and dBASE III PLUS are distinct programs that operate in very different ways and are intended to serve very different purposes. More than one well-versed dBASE programmer has been initially baffled by RapidFile, even though it is a much simpler program to learn. In an effort to help dBASE programmers quickly learn how to access RapidFile's full capability, this article lists dBASE III PLUS functions and tells how to achieve the same or similar results in RapidFile.

Some very basic functions, such as ABS(), ASC(), and INT(), are identical in the two programs. More commonly, functions such as STUFF() and DTOC() can be performed in RapidFile with the same (or a little more) effort as in dBASE III PLUS. dBASE functions that are not applicable to RapidFile (NDX(), for instance) and functions that cannot be duplicated in RapidFile have been omitted.

In the function definitions, < expC > represents a character string, < expD > represents a date, and < expN > represents a numeric value.

ABS(< expN >) The absolute value of < expN > .

In RapidFile: ABS(< expN >)

ASC(< expC >) The number representing the ASCII equivalent of the leftmost character of the < expC > .

In RapidFile: ASC(< expC >)

AT(< expC1 > , < expC2 >) A number indicating the starting position of < expC1 > inside < expC2 > . Returns zero if < expC1 > isn't there.

In RapidFile: INSTR(X, <expC2>, <expC1>)

Comments: The variable X lets you specify the starting location for the search. <expC2> can be either a string or a field name. Because the output of the function is numeric, the field type must be calculated-numeric.

BOF() Returns .T. if the record pointer is before the first logical record of the file.

In RapidFile: IFNUM(RECNO = 1, 1, 0)

Comments: Set up a calculated-numeric field as described under the dBASE III PLUS function RECNO() in this article. Returns 1 or 0 rather than .T. or .F.

CDOW(<expD>) The name of the day of the week used in <expD>.

In RapidFile: IFSTR$(<expD> = 0, "", DAY$(<expD>))

Comments: The more obvious DAY$(<expD>) will return "Saturday" for a blank date.

CHR(<expN>) A character string indicating the ASCII equivalent of <expN>.

In RapidFile: CHR$(<expN>)

Comments: Note the dollar sign in the RapidFile version.

CMONTH(<expD>) The name of the month used in <expD>.

In RapidFile: RMBLANK$ (MID$("January February
March April May June July
August SeptemberOctober November December ", MONTH(<expD>) - 1)
* 9 + 1, 9)))

Comments: It's important to note that for this to work, each section of the string for each month must be nine characters long. Thus, *May* must be followed by six spaces, *July* by five spaces, *September* by no spaces, and so on.

CTOD(<expC>) Date of the <expC>.

In RapidFile: DATE#(<expC>)

Comments: <expC> should be in MM/DD/YYYY or MM-DD-YYYY format. For a year in the twentieth century, only the last two digits of the year are necessary.

DATE() The system date in the SET DATE format.

In RapidFile: TODAY()

Comments: If entered in a noncalculated field, this outputs the date from the computer's clock when the record is added or edited. If entered in a calculated date field, it will update the date each time the file is retrieved.

The RapidFile DATE() function outputs the current date on the computer clock when any information is added or changed in a record. The date does not change when the file is loaded.

DAY(<expD>) Number indicating the day of the month.

In RapidFile: DAY(<expD>)

DISKSPACE() The number of bytes left on the default disk.

In RapidFile: **Ctrl-F,D**. The **File:Disk/directory** option causes the available disk space to be displayed on the status panel.

DOW(< expD >) Number of the day of the week used in < expD > .

In RapidFile: (INSTR(1, "Sunday Monday Tuesday WednesdayThursday Friday Saturday ",DAY$(< expD >)) + 8) / 9

Comments: It's important to note that for this to work, each section of the string for each month must be nine characters long. Thus, Sunday must be followed by three spaces, Tuesday by two spaces, Wednesday by no spaces, and so on.

DTOC(< expD >) Character string in the format "MM/DD/YY"

In RapidFile: IFSTR$(< expD > = 0, "", MID$(STR$(100 + MONTH(< expD >)), 2, 2) + "/" + MID$(STR$(100 + DAY(< expD >)), 2, 2) + "/" + MID$(STR$(YEAR(< expD >)), 3, 2))

EXP(< expN >) The value of e to the < expN > power.

In RapidFile: 2.718281828459045^ < expN > (where < expN > is the power)

Comments: Since this works only for integers, it is of limited usefulness. As it is, the base of the exponentiation is only an approximation for e.

FOUND() Returns .T. if a match was found for a previously issued search command.

In RapidFile: Use **Ctrl-S**(earch) or **Ctrl-R**(ecords),**L**(ocate).

IIF(< expL > , < exp1 > , < exp2 >) Returns < exp1 > if < expL > is evaluated to be .T., < exp2 > if < expL > is .F. (Note that < exp1 > and < exp2 > must be the same data type.)

In RapidFile: IFNUM(X, < expN1 > , < expN2 >) or IFSTR$(X, < expC1 > , < expC2 >)

Comments: Use IFNUM for numeric return variables and IFSTR$ for character return variables.

INT(< expN >) The integer portion of < expN > .

In RapidFile: INT(< expN >)

Comments: This function is undocumented.

ISALPHA(< expC >) Returns .T. if the first character of < expC > is a letter.

In RapidFile: IFNUM((ASC(< expC >) > = 65 .AND. ASC(< expC >) < = 90) .OR. (ASC(< expC >) > = 97 .AND. ASC(< expC >) < = 122), 1, 0)

Comments: Works for uppercase and lowercase letters from A to Z.

ISCOLOR() Returns .T. if the system is running in color.

In RapidFile: Handled as part of the installation process.

ISLOWER(< expC >) Returns .T. if the first character of < expC > is a lowercase letter.

In RapidFile: IFNUM(ASC(< expC >) > = 97 .AND. ASC(< expC >) < = 122, 1, 0)

ISUPPER(<expC>) Returns .T. if the first character of <expC> is an uppercase letter.

In RapidFile: IFNUM((ASC(<expC>) >= 65 .AND. ASC(<expC>) <= 90), 1, 0)

LEFT(<expC>, <expN>) A string containing the leftmost <expN> characters from the <expC>.

In RapidFile: LEFT$(<expC>, <expN>)

LEN(<expC>) The length of the <expC>.

In RapidFile: LEN(<expC>)

LOWER(<expC>) The lowercase equivalent of <expC>.

In RapidFile: CHR$(ASC(<expC>) + 32)

Comments: This works on only the first character of a string and assumes that the character is uppercase.

LTRIM(<expC>) Returns <expC> with all leading blank characters removed.

In RapidFile: RMBLANK$(<expC>)

Comments: This is a "best approximation." RMBLANK$() removes *all* the spaces from a string.

LUPDATE() Date of last update to the database file.

In RapidFile: DATE()

MAX(<expN1>, <expN2>) The value of the maximum of <expN1> or <expN2>.

In RapidFile: MAX(<expN1>, <expN2>)

MIN(<expN1>, <expN2>) The value of the minimum of either <expN1> or <expN2>.

In RapidFile: MIN(<expN1>, <expN2>)

MOD(<expN1>, <expN2>) The remainder of <expN1> divided by <expN2>.

In RapidFile: IFNUM(<expN2> = 0, -1,
<expN1> - INT(<expN1> / <expN2>) * <expN2>)

Comments: This is valid only if both numbers are integers. It returns -1 if <expN2> is 0.

MONTH(<expD>) Number of the month used in <expD>.

In RapidFile: MONTH(<expD>)

RECCOUNT() Number of records in the database file.

In RapidFile: COUNT: <fieldname>

Comments: Create a Report field with this formula. This will count only those fields that are not blank.

RECNO() Number of the current record.

In RapidFile: PREVIOUS:RECNO + 1

Comments: Create a calculated field called RECNO and place this formula in it. Or, you could use **Ctrl-N** in a numeric field in each record that you add.

RECSIZE() Number of bytes needed by a record.

In RapidFile: LEN(F1) + LEN(F2)... where F1, F2... are fields

REPLICATE(< expC > , < expN >) A string containing < expN > repetitions of the < expC > .

In RapidFile: STRING$(< expN > , < expC >)

Comments: Notice that the parameters of the function are reversed in the RapidFile version.

RIGHT(< expC > , < expN >) A string containing < expN > characters from the right of < expC > .

In RapidFile: RIGHT$(< expC > , < expN >)

ROUND(< expN1 > , < expN2 >) The value of < expN1 > rounded to < expN2 > decimal places.

In RapidFile: < expN > + .0000005

Comments: The number of decimal places displayed is determined by the File Structure Table or individual layout design.

RTRIM(< expC >) < expC > with all the trailing blanks removed.

In RapidFile: RMBLANK$(< expC >)

Comments: This is a "best approximation." RMBLANK$() removes *all* the spaces from a string.

SPACE(< expN >) A string of blanks < expN > characters in length.

In RapidFile: STRING$(< expN > , " ")

Comments: Note that there is a single space between the quotation marks.

STR(< expN1 > , < expN2 > , [< expN3 >]) The character string of < expN1 > , < expN2 > characters long with < expN3 > decimal places.

In RapidFile: STR$(expN)

Comments: Fields are formatted by the File Structure Table or individual layout design.

STUFF(< expC1 > , < expN1 > , < expN2 > , < expC2 >) Overlay < expC1 > with < expC2 > , starting at < expN1 > for < expN2 > characters.

In RapidFile: LEFT$(< expC1 > , < expN1 >) + < expC2 > + RIGHT$(< expC1, len(< expC1 >) - < expN2 > - < expN1 >)

SUBSTR(< expC > , < expN1 > [, < expN2 >]) A string that is < expN2 > characters long and starts at < expN1 > inside < expC > .

In RapidFile: MID$(< expC > , < expN1 > , < expN2 >)

Comments: Note that < expN2 > is not optional in RapidFile.

TIME() The system time in "hh:mm:ss" format.

In RapidFile: Press **CTRL** and the colon key (:).

TRANSFORM(< expN > | < expC1 > , < expC2 >) A character string created from either the < expN > or < expC1 > in the format of < expC2 > . Use @...SAY PICTURE options to format.

In RapidFile: Modify structure or Layout: Design Layout and pick from the available formats.

TRIM(< expC >) Returns < expC > with all the trailing blanks removed.

In RapidFile: RMBLANK$(< expC >)

Comments: This is a "best approximation." RMBLANK$() removes *all* the spaces from a string.

UPPER(< expC >) The uppercase equivalent of < expC > .

In RapidFile: CHR$(ASC(< expC >) - 32)

Comments: This works on only the first character of a string and assumes that the first character is lowercase. See LOWER().

VAL(< expC >) The numeric value of the < expC > .

In RapidFile: NUM(< expC >)

YEAR(< expD >) Number of the year used in < expD > .

In RapidFile: YEAR(< expD >)

RapidFile Tips

By Scott Haynes

Commas in Formatted Cells

Although RapidFile allows you to enter commas in a numeric field, numbers following the comma may be truncated when the field is formatted for custom display. For example, if the number 1,234 is entered into a numeric field and then **Field format:Fixed** is set ON with two decimal places, the number displayed in the field will be 1.00, and the field will have a value of 1 when used in calculated fields, report fields, and **Records:Totals**. This occurs because RapidFile assumes the first non-numeric or decimal character to be text, and ignores all characters from that point on. All the numbers before the text character are used as the value in calculated fields, report fields, and **Records:Totals**. Therefore, if the first character in the field is a nonnumeric character, such as "$" or "(", the entry is treated as text and it is displayed exactly as entered.

Nested Expressions

RapidFile ignores an ending parenthesis in expressions as long as there are fewer ending parentheses than beginning parentheses. Therefore, entering too few ending parentheses in an expression does not prevent the expression from functioning properly. For example,

```
IFSTR$([Fld1="F"],"Female",IFSTR$([Fld1="M"],"Male",Fld1
```

does not generate an error and does work correctly.

Display Dates in Other Centuries

Date fields are created with a default width of eight. If a date such as "01/01/2001" is entered, only "01/01/20" is displayed. This is inconsistent with dBASE III PLUS, which displays "01/01/01" and may cause confusion when working with calculated date fields that return dates from earlier or later centuries.

The alternative is to increase the width of date fields that may contain dates from centuries other than the twentieth century.

Display BC Dates

RapidFile cannot display dates below 01/01/0000 when decrementing a calculated date field. All dates below 01/01/0000 are displayed as 01/01/0000. For example, subtracting 693960 from 01/01/1900 correctly displays 01/02/0000; however, subtracting any number greater than 693961 always displays 01/01/0000.

Saving Layouts or Snapshots

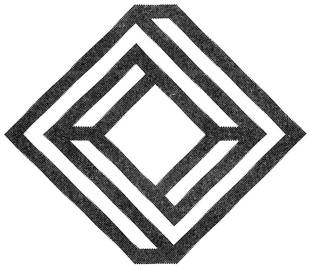
RapidFile allows you to use wildcards (* and ?) when using or saving layouts and snapshots. But, if the wildcard search does not produce a match, the wildcard character is saved as part of the name. For example, if a layout with the name of "Anyfile" already exists, attempting to save a layout with the name "Any*" produces the message

"Layout:Anyfile already exists, OK to replace? Yes No."

If a layout named "Anyfile" does not exist, but there is another layout named "Anything", the user is prompted with the message

"Layout: Anything already exists, OK to replace? Yes No."

However, if there are no saved layouts that begin with the letters "Any," a new layout is saved with the name "Any*."



ANOMALIES

DEFINITION OF ANOMALY

We use the word "anomaly" to mean any one of the following:

- INCORRECT COMMAND (or function) IMPLEMENTATION. A command or function does not execute as specified.
- INCOMPLETE COMMAND (or function) IMPLEMENTATION. A command or function does not execute as specified in every use of the command or function.
- DATA OR LOGICAL BOUNDARY LIMITATION. A command or function cannot perform an operation at a specified data or logical boundary.

  NOTE: Ashton-Tate reserves the right to withhold any further information that may reveal proprietary or trade secret information.

dBASE III PLUS

@...GET...PICTURE

Using **Ctrl-W** or **Ctrl-End** to exit an active @...GET...PICTURE of a numeric field or memory variable may produce undesirable results. When @...GET is activated, the existing number is displayed right justified. As new numbers are entered, they are displayed left justified until the display is exited, at which time the new number is right justified. However, exiting the display with **Ctrl-W** saves the digits displayed, concatenating the old number to the new. Any spaces between the numbers are ignored. This occurs only when a PICTURE clause is included. For example,

```
SET DELIMITERS ON
mvar = 10
@ 10,10 GET mvar PICTURE "999999"
READ
```

displays

```
:    10:
```

entering 100 followed by **Ctrl-W** displays

```
:100 10:
   └─ Enter Ctrl-W here.
```

This sets the value of mvar to 10010. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

APPEND FROM SDF

dBASE III PLUS stops parsing the APPEND command when the SDF option is encountered. APPEND FROM < Filename > TYPE SDF FOR < Condition > appends all records, whether or not the new records meet the FOR < condition > . However, switching the order of the TYPE and FOR options within the command line correctly filters out records that do not match the FOR < Condition > . For example,

```
APPEND FROM Test1 TYPE SDF FOR Amount < 100
```

incorrectly appends all records. But,

```
APPEND FROM Test1 FOR Amount < 100 TYPE SDF
```

correctly filters out records where the value of Amount is less than 100. APPEND FROM TYPE DELIMITED FOR < Condition > correctly filters records regardless of the option order. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

COPY TO

If field names are duplicated in the FIELDS list of the COPY TO command, each field in the list, including duplicates, becomes a field in the resultant database file. All fields can be EDITed and BROWSEd but, only the first field with any name can be accessed with @...SAY/GET, ?/??, DISPLAY, and LIST.

When the new file structure is edited with MODIFY STRUCTURE, the error message "Field name already in use." is produced and the edit must be aborted with **Ctrl-Q** or **Esc**, unless the duplicate field names are immediately changed. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

CREATE VIEW

Opening a view file that was created with CREATE VIEW does not automatically SET FIELDS ON unless one of the fields was deselected from the CREATE VIEW **Set Fields** list when the .VUE was created or unless SET FIELDS ON was issued prior to CREATE VIEW. When creating a view file, all fields are selected by default. When the .VUE file is saved, the status of SET FIELDS is saved with it. If SET FIELDS was OFF prior to CREATE VIEW, it is not changed unless a field is deselected from the **Set Fields** fields list. For example,

```
SET FIELDS OFF
CREATE VIEW Test
* ---Select a database file and save the .VUE file.
SET VIEW TO Test
SET FIELDS ON
LIST
```

displays the record numbers only. DISPLAY STRUCTURE reveals that no fields are selected. However, MODIFY VIEW <Test> shows that all fields are selected.

If more than one database file is opened by SET VIEW TO and a relation is established, fields from database files not in the current work area are available only if SET FIELDS is ON and the fields are in the SET FIELDS TO list. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

DO CASE

When executed, the word END in a DO CASE...ENDCASE structure terminates program execution as if RETURN were encountered. When encountered outside a DO CASE...ENDCASE construct, END produces the error message "Unrecognized command verb." [16]. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

JOIN

JOIN WITH <Alias> TO <Filename> creates a new file with the structure of the database file in the current work area exclusively when the WITH file contains illegal characters (such as +, -, or &) in the filename. When valid filenames are used, JOIN creates a new file whose structure contains all the fields from the two files, provided that the new structure does not exceed the 128 field limit. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

READKEY()

The value returned by READKEY() is reset to 271 or 15 after executing a RETURN, SUSPEND, or CANCEL command. The value returned by READKEY() should remain unchanged until another full-screen command is executed. However, RETURN, SUSPEND, and CANCEL reset the key value to 271 or 15. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

REPORT FORM

REPORT FORM with SET ALTERNATE TO <Filename> and SET ALTERNATE ON produces inconsistent page breaks when the NOEJECT option is included or **Options:Page eject before printing** is "Yes." To work around this problem, SET ALTERNATE OFF before running the report and send the report to a text file by including the TO FILE <Filename> option in the REPORT FORM command line. This occurs in both versions 1.0 and 1.1 of dBASE III PLUS.

RAPIDFILE

Match Criteria Limit

Attempting to use the **Records:Locate:Matching records** or **Select:Matching records** options with more than 51 match criteria causes the system to hang.

Updating Preset Layout Settings

Page 7-2 in *Using RapidFile* states that "The preset format and size are used when you choose **Display all fields** to bring all fields back into a layout." When the **Ctrl-U** option is used to update the Preset Layout Settings in the File Structure Table, RapidFile does not use these new settings when creating any new layout that is not a table. However, if the field format is changed through the **File:Modify structure** option, all new layouts do reflect the change.

Custom Formats in New Layouts

When designing new quick report, form, and split table/form layouts, fields that are included in the layout by default do not reflect any custom formats that are set through the File Structure Table. However, new fields that are added to the layout with **Ctrl-Ins** or the **Fields:Show fields** option do reflect the custom format settings.

Selecting the **Fields:Display all fields** option while in the **Layout:Design layout** clears the default layout and redisplays all fields using the custom formats.

NOW$()

Page 13-13 of *Using RapidFile* states that when the NOW$() function is used in a calculated field, the current time from the computer clock is displayed each time the record is edited. When the field is edited and **Esc** is pressed to abort the edit, the time should not be updated. However, if data has been changed before **Esc** is pressed, the field containing the NOW$() function is updated with the current time.

FRAMEWORK II

FWSETUP

When FWSETUP is in a subdirectory other than the one containing the FW.EXE and FW.OVL files and Framework II is executed from that subdirectory, the error message

```
Please insert System Disk 2 in Drive A:
       (press ESC to continue)
```

is displayed. Pressing **Esc** has no effect; you will need to reboot to clear the error condition.

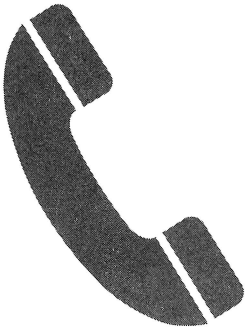
Note: The directory containing FWSETUP is considered the system path by Framework II, which searches for the system files (except FW.EXE) along that path. Therefore, if the FW.OVL file is not in that path, Framework II will continue to prompt for System disk 2.

Bootable System Disk

When using the SETUP program, Framework II system disks cannot be made bootable with DOS 2.x.

The work-around is to format a blank diskette with the /S option and use the DOS COPY *.* command to copy files from the System 1 disk to the new disk. The SETUP program can then be run from the new disk.

ON-LINE SERVICES



BY STEVE SILVERWOOD

Do you have a modem? Do you subscribe to CompuServe or The Source? If so, Ashton-Tate offers technical support on all of these electronic information services. You can quickly access the kind of quality technical information that you have come to expect from ***TechNotes***, and you can do it through your computer system. To take advantage of this progressive support method, you must subscribe to one of the services listed below. There are no additional charges for using our on-line support services, apart from the normal connect-time fees for each service. You can get specific subscription information by calling one of the following numbers:

Service	Subscriber Information	Command to find Ashton-Tate
CompuServe	(800) 848-8199	GO ASHTON
The Source	(800) 336-3366	ATSIG

To give you an idea of support that is happening on-line, the following are regular excerpts from the Ashton-Tate Forum on CompuServe. This monthly material comes from messages our Forum users have posted on the bulletin board. All message headers have been removed to protect the identities of our users. Every other message within a single conversation topic has been italicized to enhance readability.

dBASE III PLUS

While doing some serious debugging (using an ON ESCAPE DO routine that allows me to CLEAR ALL and CANCEL or SUSPEND to look at status/memory/etc., I've gotten myself into a situation where every command returns "***INTERRUPTED***" until I QUIT and restart dBASE III PLUS. I can't for the life of me figure out what I'm doing to get myself into this endless loop. I also can't seem to force it on purpose.

I do know that opening too many files will force one to return to DOS, making it necessary to restart dBASE. What causes an endless ***INTERRUPTED*** message? I know that as soon as someone explains this one, I'll suddenly remember that I knew that.

Why bother with ON ESCAPE DO...? That's what happens when you press **Esc** *with no ON ESCAPE command. Remember to type CANCEL after you are finished looking at your variables, or you will run out of file handles very quickly. I would get rid of the ON ESCAPE—that may fix your problem.*

It was originally there for running dBCODEd programs so that the **Esc** key would be recognized. It also opens a window with more instructions for the operator than dBASE would supply. When I gave up on dBCODing, I left it in.

Actually, I believe that if you SUSPEND, you must RESUME prior to a CANCEL.

No, you do not have to RESUME prior to a CANCEL.

.....

I'm using a database file that's INDEXed on STR(Entry), where Entry is a three-digit numeric field. This does not work properly unless I explicitly specify the length in the STR function call:

```
INDEX ON STR(Entry,3) TO xxxx
```

The documentation says that STR() returns a length of 10 if not specified; but I think if the database file is empty when the INDEX command is issued, then the length of STR() is zero.

This is normal operation. The length parameter must be specified when indexing.

.....

One of my users sent me this message. I had asked for a copy of the actual code, but this is what they sent me:

"Reference is to *Ashton-Tate Quarterly* April/May/June 1986 issue page 33. A program for testing the INKEY() function is shown in Figure 3. When this program is run on an IBM PC-XT and PC-AT with dBASE III PLUS, it does not work consistently with the **Leftarrow** key."

Can you give me some help? I don't have the magazine, and I haven't heard about the problem before.

That article was just a description of how to use the INKEY() function, but there is a limitation when using the **Leftarrow** *key.*

INKEY() traps **Leftarrow** *differently in version 1.1 than it did in 1.0. Specifically, in 1.0 with SET ESCAPE OFF, INKEY() returns a value of 19. With SET ESCAPE ON,* **Leftarrow** *suspends operation like* **Ctrl-S**. *In 1.1,* **Leftarrow** *is occasionally trapped by INKEY(), but more often it suspends operation regardless of the status of SET ESCAPE. If* **Esc** *is pressed after* **Leftarrow** *suspends operation, INKEY() returns a value of 19, all other keys return 0.*

As an alternative, you might consider the INKEY.BIN routine published in the April 1987 issue of **TechNotes**. It's in DL2, named INKEY.ARC.

.....

Is there a way (other than a command file) to combine logical and character fields in a LABEL or REPORT format? The construction:

```
<Logical memvar> + <Character memvar>
```

is usually rejected. Where it is accepted, as in the report, it does not give the expected results. My objective is to get a printed result from dBASE III PLUS such as:

```
Logic_var = .T.
```

In other words, is there a way to convert logic variables to character variables within a report?

You can use the IIF() function as the contents of a report column, as in:

```
"Logic_var = " + IIF(logical,"TRUE", "FALSE")
```

.....

Help! I am looking for a utility program that will convert files from Lotus 1-2-3 release 2.1 to dBASE III PLUS version 1.1, and then back again. Do you know where I can find one? Any help will be appreciated.

You already have the ability to do that with Lotus 2.1 and dBASE III PLUS. Since dBASE III PLUS can't read .WK1 files directly, you have to create an ASCII or .PRN file in Lotus. Do all the same things you would do in printing the Lotus file to a printer, but when the Lotus status bar under /P gives you the two options PRINTER and FILE choose the file option. A file with the name you assign to it and the extension .PRN will be created.

In dBASE III PLUS, create a database file with a field that corresponds to each column in your Lotus file. The fields must be the same length as the particular column(s) in your Lotus file. Be sure to define the numeric fields with the correct decimal points, and so on.

Once that is done, when you are in dBASE III PLUS, simply USE the dBASE file you have created, APPEND FROM < Lotus file name.PRN > SDF, and the data will be read into dBASE.

.....

Is it possible to build search strings right into the Report file so that when the Report file is called up and executed it will conduct the search each time without further commands? Also, can this report include instructions to use certain indexes and ignore others automatically?

If this can be done it will make it easy for the staff to run the various printouts they need in an efficient manner.

Retrieval specifications are not part of the REPORT FORM; rather, they are part of the command used to invoke the report, such as:

```
REPORT FORM Filename FOR Field="expression"
```

To make it simpler for the people in the office to run various reports, why not set up a simple program shell in dBASE that prompts them with a menu of various report choices, allowing them to select one at will?

Index files are a little different. While you can have up to seven index files open at one time with a file, only the first one is used to sequence or locate the data. Therefore, if you give the command:

```
USE Filename INDEX Index1, Index2, Index3
```

all three index files will be active, and will automatically be updated to reflect any changes made to the database files. If a SEEK or FIND command is given, though, or if the file is LISTed out, it will come out in order based on Index1's key expression only. The rest are there specifically to allow automatic updating, and that's all.

Once again you have come to my rescue. I'll give it a try. That will probably mean I'll be back with questions about setting up menus.

.....

I am having a small problem that I can't seem to solve. Recently I wrote a program to add members to one of our membership database files. Included in this program was a screen that I had written. Everything appeared to work fine. Tonight I wrote a main menu program for this file in which one of the selections calls up the Add program. However, when the screen appears the first line, which is a title, gets chopped about in half. Actually, the full line appears for a second, then the last half disappears. The line is suppose to read "Vietnam Helicopter Pilots Association Membership Record" but what ends up staying on the screen is "Vietnam Helicopter Pilots Assoc". (The quotes are just for this message and aren't part of the screen.) This happens even if I exit dBASE III PLUS, reenter and try to DO the Add program without activating the main menu. Everything else appears to work, though I am not sure if the indexes are still being activated. Anyone got any ideas?

By default, the SCOREBOARD is SET on. If you are writing to the very first row of the screen, row 0, then it is possible that the scoreboard is overwriting your title. This will only happen if STATUS is SET off.

.....

RapidFile

I have just completed creating two files in RapidFile. One is for daily entry of clients and their purchases. This file does not include the client address on each record. The second file is for client address information. It is to be used for mailings. I want to bring sales totals from the first file, grouped by client name, into the client record of the second file. Can this be done with RapidFile? I thought that the APPEND command was the answer, but I can't seem to make it happen.

From the sound of it, you'd need to establish a relation between the two files to perform the update. RapidFile does not provide for relating two files, nor does it allow updating of one file from the contents of another.

APPENDing the data will just create new records, not update existing ones.

I hate to say it, but RapidFile isn't going to do it for you that way.

Thank you. I think in the back of my mind I already knew that I needed dBASE III PLUS to do what I want to do. You have an excellent manual with RapidFile and if I were able to do this, I'm sure it would have been in the manual. I have dBASE III PLUS on the shelf (what a waste, right?). I have found it too difficult to set up with the limited time I can spend with it. I guess I'll have to get it out, set it up, and start studying.

.....

Framework II

I need a formula to sum a database column from within the column. **@sum**(database.col) includes its own value in the total and dividing that by two yields very strange results.

As you found out, the entire column will be summed, including the cell containing the sum. The recommended technique is to put the @sum formula on an empty word or spreadsheet frame. The database and spreadsheet can then be placed in a containing frame. This will appear exactly the same in a printed report, and will be more reliable and flexible. A database field, by definition, means that the data is all of the same kind. If you want a matrix of cells that can contain anything you want, rather than follow rules of order, you should use a spreadsheet.

.....

Is there any way to switch the **CapsLock** key's light on or off?

I've "added in" a typing tutor program that turns **CapsLock** on without affecting the key's light. It works well, but if I forget to turn **CapsLock** on before returning to Framework II, I find the CAPS indicator and capitalization on although the key's light is off. Pressing the key turns Framework II's indicator and capitalization off, but leaves the key glowing deceitfully. I found the DOS command to slow my AT-clone down for the typing tutor and then speed it back up when that program is finished, but I can't find anything in DOS or Framework II to let me tinker with the **CapsLock** key's light.

There are some functions in the Framework II Developer's Toolkit that allow you to turn **NumLock** *and* **CapsLock** *on and off, but that is independent of the keyboard lights. Each is controlled separately, and I have not seen a program that properly affects the lights. There are BIOS calls to control the lights. Perhaps, with the aid of the Developer's Toolkit, you could write a low-level function to do what you want. You might also look in the IBMHW forum for a prewritten utility.*

Thanks for the information. We're going to get the Toolkit soon, so I'll wait and see what it provides before wandering into the wilds of the IBMHW forum.

.....

I am having trouble with my extended memory with Framework II. Whenever I use any RAM-resident utility (Word-Finder, Lightning, SideKick!, etc.) and have it loaded in RAM (even on a RAM disk) Framework II will recognize only the memory I have on my motherboard (and only up to 512K). In fact, I can't activate any RAM-resident programs in Framework II since I got my IBM AT.

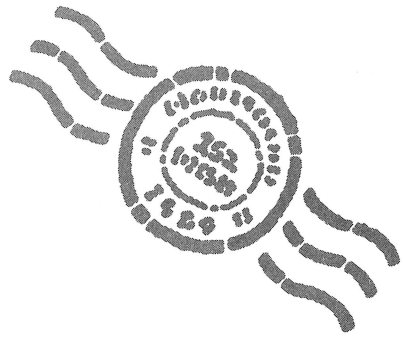
I have an IBM AT, EGA color card, 1 MB of memory on board, MTcard multifunction card w/2.5K extended memory (this card recognizes only up to 640K on motherboard so I had to configure to recognize only 512K on motherboard and 2.5K on expansion card.) My setup in Framework II is Expanded Memory on, with the AST Advantage driver installed (it worked best of all listed drivers.) I cannot keep overlays in memory because when Framework II boots up it gives me an error message — "not enough normal memory to keep overlays in memory."

If anyone has had a similar problem and figured it out could you please drop me a hint as how to correct the problem?

With 512K, Keep Overlays and Extended Memory are mutually exclusive. ALL 512K will be required to have Framework II running with extended memory. There will only be room for the tiniest of TSRs.

.....

LETTERS



Avoiding .DBF Corruption

Dear *TechNotes*:

When I list my dBASE III PLUS file, a number of records are often missing or distorted. This happens regardless of which drive I use. The program is maintained on a hard disk, with data output to floppies. Can you suggest a cure?

JT

Dear JT:

The problem you describe is probably caused by corrupted data in the .DBF file. To fix the problem, try running the DOS CHKDSK/F utility to look for lost/cross-linked clusters. Fixing the lost/cross-linked clusters on the disk often solves the problem.

Once you locate the corrupted records, you may want to recover the usable portion of the .DBF file. To do this, copy the ranges of good records to temporary database files using COPY TO ... FOR RECNO() > = ss .AND. RECNO() < = ee, where ss and ee are the starting and ending record numbers of the good records. When you finish copying the good records, USE the first temporary file and APPEND FROM the others.

Here are some suggestions that will help you avoid this problem in the future:

1. If your system has 384K or more of RAM, include the following lines in the CONFIG.SYS file:

FILES = 20
BUFFERS = 15

If your system has less than 384K of RAM, the recommended settings are:

FILES = 20
BUFFERS = 4

Check that this file is in the root directory of the disk that you boot DOS from.

2. Ashton-Tate recommends that you remove any memory-resident utilities while you run dBASE III PLUS, as they may interfere with the normal execution of the program andcould possibly cause data corruption.

3. Do not swap disks while .DBF files are open. This can permanently damage the file-allocation table. Make sure you close all files before you swap disks.

4. Do not reset your computer or turn it off while any .DBF files are open.

Adding a +1 to a Common Numeric Field

Dear *TechNotes*:

I have three .DBF files, each of which contains from 35 to 45 fields. There is a common numeric field in all three files, plus each file contains a lastname and a firstname field. They are indexed on lastname + firstname. There is no duplication of people in the three files.

How can I add a "+1" to the common numeric field based on a logical field in a fourth file? This fourth file contains some names from all the previous three; but it is not exhaustive.

JB

Dear JB:

Probably the simplest way to add a +1 to a common numeric field in three files based on a logic field in a fourth file is to SET RELATION TO the Lastname + Firstname (your index key) INTO each one of the files and then scan this fourth file and REPLACE FOR all the records that have a .T. (or .F.) value for the logical field.

The following sample code will give you a better idea of the procedure:

```
SELECT 1
USE File4
SELECT 2
USE File1 INDEX Name1
SELECT 3
USE File2 INDEX Name2
SELECT 4
USE File3 INDEX Name3
SELECT 1
SET RELATION TO Name INTO File1
SELECT 2
SET RELATION TO A->Name INTO File2
SELECT 3
SET RELATION TO A->Name INTO File3
SELECT 1
DO WHILE .NOT. EOF()
   IF Logical
      SELECT 2
      IF .NOT. EOF()
         REPLACE Numeric WITH Numeric + 1
      ENDIF
      SELECT 3
      IF .NOT. EOF()
         REPLACE Numeric WITH Numeric + 1
      ENDIF
      SELECT 4
      IF .NOT. EOF()
         REPLACE Numeric WITH Numeric + 1
      ENDIF
      SELECT 1
   ENDIF
   SKIP
ENDDO
```

Making Linear Regression Work

Dear *TechNotes*:

I am trying to use the linear-regression program in *Advanced Topics Framework II*, page 103. I typed it in exactly as it appears but couldn't get it to work. Can you provide the correct code along with instructions on how to use it?

MR

Dear MR:

The code you are referring to has two typographical errors. The first one appears on the fourth line of page 103:

```
@while((t1 := @get(@itemf1)) <> #null!,
```

The correct code for that line should be:

```
@while((t1 := @get(@item1)) <> #null!,
```

The second typo is on the first line on page 104:

```
t3 :+ sumxy - (sumy * avgx),
```

It should read:

```
t3 := sumxy - (sumy * avgx),
```

After you make these corrections, the program will run as expected. To use this program, you need to reference a spreadsheet with the data you want to use for the regression. For instance, create a containing frame called linear regression with a subframe called prog. Enter the formula

```
@linear regression.prog(region1,region2,region3)
```

in the formula area of the linear regression frame. Region1 is the region that contains the data for the x values of the regression, region2 contains the y values, and region3 will be used by the program to return the slope and the intersection of the linear regression. The entire reference to the spreadsheet must be entered (such as SS.A1). Enter the program code in the formula area of prog and recalculate the linear regression frame.

Cumulative Index

January 1987 through August 1987

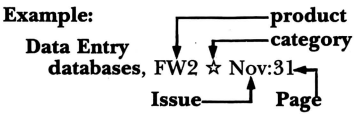
Products:

D2	-	**dBASE II**
D3	-	**dBASE III**
DR	-	**dBASE III Developer's Release**
DP	-	**dBASE III PLUS**
DPA	-	**dBASE III PLUS ADMINISTRATOR**
DB	-	**dBASE Products**
FW	-	**Framework**
FW2	-	**Framework II**
MM	-	**Multimate**
RF	-	**RapidFile**
UT	-	**Utilities**

Categories:

☆	-	**Articles**
¥	-	**International**
□	-	**Letters**
○	-	**On-Line Services**
◀	-	**Sample Programs**
§	-	**Usage Tips**

Example:



@...GET, DP ○ Apr:59
@...SAY, DP ○ Apr:59, DP □ Apr:64, D2 ¥ May:67
  to text files, DP ☆ May:3
  using drivers, DP ☆ May:12
  nondestructive cursor positioning, DP § Jul:28
  page eject, DP § Jul:24
  to text files, DP ☆ Jun:77
@...SAY...PICTURE
  with negative numbers, DP § Jun:27

A

@abs, FW2 ☆ Mar:26
ACCESS, DPA § Feb:50
Adding users on a network, DPA ☆ Apr:12
Addlabel.FW2, FW2 ◀ Aug:49
ADDUSER.COM, DPA ☆ Apr:3, DPA ☆ Apr:12
Add_lf.FW2, FW2 ◀ Jan:46, FW2 ○ Jul:66,
  See also: Telecommunications
Adjust.FW2, FW2 ◀ Aug:51,
  See also: Addlabel.FW2
Adjusting Field Width, RF § Jul:53
Alt-1.FW2, FW2 ◀ Jul:44
Alt-B.FW2, FW2 ◀ Jul:46
Alt-C.FW2, FW2 ◀ Jul:46, FW2 ◀ Aug:52
Alt-F.FW2, FW2 ◀ Jul:47
Alt-K.FW2, FW2 ◀ Jul:43, FW2 ☆ Aug:49
Alt-V.FW2, FW2 ◀ Jul:47
ALTERNATE = in CONFIG.DB, DP ☆ Jan:22
APPEND, DP § Jan:44, DP ☆ Mar:13, FW2 ○ Mar:61, RF ○ Aug:69, DPA □ Aug:73
  correctly updating indexes on a network, DPA § Jul:22
  to a DELIMITED file, DP ○ Jun:70
APPEND FROM, DP ○ Jul:63
  delimited, DP § Aug:43
  SDF, DP § Feb:49, DP ○ Feb:57, DP ○ Apr:57
Arrays
  imitating in dBASE III PLUS, DP § Jul:26
  save and restore, UT § Mar:54
ASC(), DP ☆ Apr:16
ASSIST, DP § Apr:35, DP § Apr:41, DP § Jun:27
AT&T 7300, DP □ Jul:69
AT(), DP □ Feb:63, DP ☆ Jul:14
ATO
  with CompuServe, FW2 ○ Jun:69
ATTRIB, DPA ☆ Apr:15
attributes, FW2 § Aug:53
AUTOEXEC.BAT, DP ○ Jan:58
@avg, FW2 ☆ Feb:10

B

BACKUP.EXE, DP ☆ Jul:3
@beep, FW2 ○ Jan:59
BELL = in CONFIG.DB, DP ☆ Jan:22
Break character, FW2 ☆ Jul:39
BROWSE, DP § Jan:45
  EGA 43-line mode, DP ○ Feb:56
  WIDTH, DP § Feb:47
BUCKET = in CONFIG.DB, DP ☆ Jan:21

C

Calculated fields, RF ☆ Jun:77
CALL, DP ☆ Jan:28, DP ☆ Feb:21, DP ☆ Mar:19, DP ○ Apr:59
CANCEL, DP ○ Aug:66, DP ○ Aug:67
Capslock key, FW2 ○ Aug:70
CARRY = in CONFIG.DB, DP ☆ Jan:22
Case folding, DP ¥ Jan:61
CATALOG = in CONFIG.DB, DP ☆ Jan:22
Catalogs, DP □ Jan:64, DP □ Jul:68
CENTURY = in CONFIG.DB, DP ☆ Jan:22
CHANGE, DP ○ Apr:59
Chgtype.PRG, DP ◀ Aug:29,
  See also: Memojoin.PRG
Chkfiles.PRG, DP ◀ Aug:26,
  See also: Memojoin.PRG
Chkstat.PRG, DP ☆ Aug:38,
  See also: Getstat.PRG
CHMOD, UT ☆ Apr:15
@chr(), FW2 ☆ Mar:25
CHR(0), DP ☆ Jun:5
CHR(138), DP § Aug:43
CHR(141), DP § Aug:43
CLEAR ALL, DP ○ Aug:66
CLOSE PROCEDURE, DP § Apr:40
Collation sequence, DP ¥ Jan:61
Color
  blinking, DP § Jun:28
  high-intensity, DP § Jun:28
COLOR = in CONFIG.DB, DP ☆ Jan:22
Columnar listings, DP § Aug:41
COMMAND = in CONFIG.DB, DP ☆ Jan:21, DP § Apr:35
Commas
  in formatted cells, RF § Aug:60
Comments, DP ○ Jul:62
Compaq 386, DP § Feb:47
  DOS, UT § Apr:38
CompuServe
  downloading articles from, ○ Aug:3
COMSPEC, UT § Apr:39
CONFI256.SYS, DP □ Feb:64
CONFIG.DB, DP § Feb:47, DP □ Feb:64, DP □ Mar:70, DP § Apr:35, DPA § Jul:20
  classes of commands, DP ☆ Jan:20

configuration commands, DP ☆ Jan:21
creating and editing, DP ☆ Jan:20
function keys, DP ☆ Jan:25
set commands, DP ☆ Jan:22
setting colors, DP § Jul:28
CONFIG.SYS, DP ☆ Jan:40, DP □ Feb:64, UT § Apr:40, DP § Jun:29, DP □ Aug:70
BUFFERS, DP ☆ Jan:19, DPA ☆ Aug:6
FILES, DP ☆ Jan:19
CONFIRM = in CONFIG.DB, DP ☆ Jan:22
CONSOLE = in CONFIG.DB, DP ☆ Jan:22
CONTINUE, DP § Apr:41
Converting
numbers to words, DP ☆ Mar:17
COPY, FW2 ☆ Jan:9, DP § Jan:45, FW2 ○ Mar:62, FW2 ☆ Apr:22, FW2 ○ Apr:60
absolute cell references, FW2 ☆ Mar:29
boldface text, FW2 ☆ Mar:29
frames, FW2 ☆ Mar:29
soft hyphens and page breaks, FW2 ☆ Mar:30
STRUCTURE EXTENDED, DP § Mar:52
Cpsep.PRG, DP § Jan:45,
See also: Exporting
CREATE, DP § Mar:52
CREATE LABEL, DP § Feb:48
CREATE SCREEN, DP ☆ Mar:10
Ctrl-F10, FW2 ☆ Jul:37
Ctrl-Return, FW2 ☆ Jan:9
Ctrl-S, FW2 ☆ Jul:39
Current system date, DP ○ Jun:70,
See also: Serial.PRG
Cursor pointing, FW2 § Jun:42
Cursor positioning
nondestructive, DP § Jul:28
Cut and Paste, FW2 § Jul:46, FW2 § Aug:53

D

Data entry, FW2 ☆ Feb:10
field ditto, DP ☆ Mar:13
Data Terminal Ready (DTR) signal, FW2 ☆ Jul:36
DATE(), DP ¥ Jan:62
Dates
conversions, DP ○ Mar:61
display BC, RF § Aug:60
display from other centuries, RF § Aug:60
importing from dBASE III PLUS, FW2 § Aug:52
logical comparisons, DP § Jul:24
test for blanks, DP □ Mar:69
using in match criteria, RF § Jul:54
DATES.ASM, FW2 ◀ Jun:35,
See also: Mymain.ASM
Day.C, FW2 ◀ Jun:35,
See also: Mymain.ASM
Day.FW2, FW2 ◀ Jun:37,
See also: Mymain.ASM
dBASE Administrator
correctly updating indexes with APPEND, DPA § Jul:22
creating a shared directory, DPA ☆ Apr:5
DISPLAY STATUS, DPA ○ Apr:57
DISPLAY USERS, DPA § Apr:34, DPA ○ Apr:57
EDIT, DPA § Jul:22
files to erase when reinstalling, DPA § Aug:46
IBM problems, DPA § Apr:34
modifying network parameters, DPA ☆ Aug:8
Novell network MAP command, DPA § Aug:45
problems on 3COM 3+ version 1.1, DPA ○ Apr:57
protect database re-encryption, DPA § Apr:34
running from a workstation, DPA ☆ Apr:14
dBASE III PLUS LAN PACK
Novell network, DPA □ Aug:73
dBASE Programmer's Utilities, DP ○ Jul:65
@dbasefilter, FW2 ○ Feb:58, FW2 § Aug:53
dbCODE, DP § Aug:44, DP ○ Aug:66
DBF header structure, DP ☆ Jul:6
DBF(), DP □ Jan:64
DBNETCTL.300, DPA ☆ Apr:3, DPA ☆ Apr:13
DBSYSTEM.DB, DPA ☆ Apr:15
DCONVERT, DP ○ Jan:57, DP ○ Apr:58
DEBUG.COM
general syntax, DP ☆ Jul:3
internal registers, DP ☆ Jul:5
memory addresses, DP ☆ Jul:4
to change the first byte of a .DBF file, DP ☆ Jul:7
to locate EOF markers embedded in files, DP ☆ Jul:7
to recover from a DOS BACKUP without RESTORE, DP ☆ Jul:7
DEBUG = in CONFIG.DB, DP ☆ Jan:23
DECIMALS = in CONFIG.DB, DP ☆ Jan:23
DEFAULT = in CONFIG.DB, DP ☆ Jan:23, DP ☆ Mar:9
DELETED(), DP ○ Jul:63
DELETED = in CONFIG.DB, DP ☆ Jan:23
DELIMITED file
APPEND to, DP ○ Jun:70
DELIMITERS = in CONFIG.DB, DP ☆ Jan:23
Design Layout, RF § Jul:54
DEVICE = in CONFIG.DB, DP ☆ Jan:23
@diffdate, FW2 § Apr:31
DIR, DP ☆ Mar:9, DP § Aug:44
DIRECTORY, DP § Aug:44
DISPLAY, DP § Jun:29
@display, FW2 ○ Jan:59
DISPLAY STATUS, DP ○ Mar:63, DPA ○ Apr:57, DPA § Jul:20
DISPLAY USERS, DPA ☆ Apr:15, DPA § Apr:34, DPA ○ Apr:57
Disptest.PRG, DP ◀ Jun:17,
See also: Newsave.ASM
Ditto.PRG, DP ◀ Mar:14,
See also: Data entry
DO CASE
using arrays to replace, DP § Jul:27
DO...WITH, DP § Jul:24
DOS access, FW2 ☆ Jan:15
prompt change, FW2 § Apr:32
DRAG, FW2 ☆ Mar:31
Drive cabinets, FW2 § Jun:43
Drive settings, FW2 ☆ Jan:8
Dual column macro, FW2 ◀ Apr:23,
See also: Printing with Framework II

E

ECHO = in CONFIG.DB, DP ☆ Jan:23
EDIT, DP ○ Apr:59
minimizing locked time on a network, DPA § Jul:22
Editing
display hidden characters, FW2 ☆ Mar:31
documents, FW2 § Jan:46
Undo, FW2 ☆ Mar:31
EDIX, DP ☆ Jan:36
EGA,
See also: Newsave.ASM
43-line mode, DP ☆ Jun:15
Empty.PRG, DP ◀ Feb:28,
See also: Word wrapping
End of Line Char Count, FW2 § Jul:42
End-of-file markers
using DEBUG to repair, DP ☆ Jul:3
ENDSPOOL
using on the Novell Advanced NetWare network, DPA § Jul:20
EOF(), DP § Apr:38
ERASE
field, DP ○ Jun:73
Error messages in Assembler
"Warning: no stack segment", DP ☆ Jul:16
Error messages in dBASE III PLUS
"*Execution error on +"**, DP ☆ Feb:26
"*Execution error on SUBSTR(): ..."**, DP § Apr:40
"*INTERRUPTED***"**, DP ○ Aug:66
".DBT file cannot be opened", DP § Feb:48
"ALIAS name already in use", DP § Aug:44
"Alias not found", DP ○ Feb:57, DP ○ Mar:63
"Cannot erase a file which is open", DP § Jul:25
"CONTINUE without LOCATE", DP § Apr:41
"Do you wish to delete field", DP ☆ Mar:10
"File does not exist", DP ○ Jun:70
"File is already open", DP § Apr:42
"File not found", DP ○ Jun:73
"Illegal value", DP § Feb:47
"Internal Error Illegal OP Code", DP ○ Jul:64
"Invalid DIF vector - DBF field mismatch", DP § Apr:38
"Invalid printer port", DP § Apr:41
"NET 803: Network path not found", DPA ☆ Mar:10
"Not a dBASE database", DP ☆ Jul:8
"not readable", DP ☆ Mar:9
"Numeric overflow (data was lost)", DP § Feb:49

"Printer is either not connected or turned off", DP § Apr:41
"Printer not ready", DP ☆ Mar:11, DP § Apr:41
"Syntax error in field expression", DP ○ Jan:57
"Syntax error", DP ○ Mar:63, DP ○ Jul:63
"Variable not found", DP § Jan:45, DP ○ Mar:63
Error messages in Framework II
"Frame badly formatted - unable to read it", FW2 ☆ Mar:25
"Missing operand", FW2 § Aug:53
"Not able to load extended memory driver", FW2 ☆ Jan:10
"not enough normal memory to keep overlays...", FW2 ○ Aug:70
"Not enough normal memory to use extended memory", FW2 ☆ Jan:11
"Read error while trying to sync disk", FW2 ☆ Jul:40
"Telecomm now inactive", FW2 ☆ Jul:37
"Waiting for trap character", FW2 ☆ Jul:38
Error messages in RapidFile
"Current memo too large...", RF § Jun:51
"File <filename> not found.", RF § Jul:54
ESCAPE= in CONFIG.DB, DP ☆ Jan:23
European versions, DP ¥ Jan:60
EXACT= in CONFIG.DB, DP ☆ Jan:23
Exporting
ASCII file, DP § Jan:45
saving data to a text file, FW2 ¥ Apr:56
SYLK files, FW2 ☆ Mar:38
Expressions
nested, RF § Aug:60
EXTEND SELECT, FW2 § Jan:46, FW2 ○ Jan:58
Extended memory, FW2 ☆ Jan:10, FW2 ○ Aug:70
External programs
interfacing dBASE III PLUS, DP □ Mar:70

F

Fields
adding to common numeric, DP □ Aug:72
changing names, RF § Jun:50
totaling, RF § Jun:51
File corruption, DP □ Aug:70
File handles, FW2 ☆ Mar:30
File to load on startup, RF § Jun:50
File transfers, FW2 ☆ Jul:41
Files
saving, RF § Jun:51
Fill.PRG, DP ◀ Mar:54,
See also: CREATE
Fillcol.FW2, FW2 ◀ Aug:50,
See also: Addlabel.FW2
Filtering, FW2 □ Mar:71
Floating dollar signs, DP § Apr:39
REPORT FORM, DP § Aug:41
FOR, DP § Apr:41
Foreign language translation
FRED programs, FW2 ☆ Jun:39
Formal definition of a keyword, FW2 § Apr:31
Forms view, FW2 ☆ Mar:32
Framelock, FW2 □ Jan:63
Frames
creating sized, FW2 § Jun:43
size, FW2 ☆ Jan:12
Framework II Developer's Toolkit
Numlock and Capslock keys, FW2 ○ Aug:70
using with Microsoft C, FW2 ☆ Jun:30
Function keys
reprogramming to recognize
DOS functions, DP □ Apr:64
FUNCTION= in CONFIG.DB, DP ☆ Jan:25
Functions
comparing dBASE III PLUS and RapidFile, RF ☆ Aug:54
using in match criteria, RF § Jul:53

G

GETENV(), DP § Apr:39
GETS= in CONFIG.DB, DP ☆ Jan:21
Getstat.PRG, DP ☆ Aug:35
Graphics library
save and restore arrays, UT § Mar:54
scaling, UT § Mar:55
Graphing, FW2 ☆ Mar:32
GraphLink
graphics integration, MM ☆ Jan:41

H

Headers
in Quick Reports, RF § Jun:50
HEADINGS= in CONFIG.DB, DP ☆ Jan:23
HELP, FW2 ☆ Jan:3, FW2 ☆ Mar:32
HELP= in CONFIG.DB, DP ☆ Jan:23
@hide, FW2 ☆ Mar:31
HISTORY= in CONFIG.DB, DP ☆ Jan:23
HP LaserJet, MM ☆ Apr:26, MM § Apr:43, MM ○ Jun:75,
See also: Printing with MultiMate
line and box drawing with, MM ☆ Jun:52
printing to, RF § Jun:50
soft fonts, MM ○ Jun:75
HP-3000, DP □ Jul:69

I

IBM PC Network Program
fine-tuning, DPA ☆ Aug:5
IBM Token-Ring Network
with MultiMate, MM ○ Jun:75
ID, FW2 ☆ Jan:4
IDLAN, DPA ☆ Apr:10
IIF(), DP § Apr:40, DP ○ Jul:64
Importing
ASCII, MM § Jun:58
comma separated text file, FW2 ○ Apr:60
dBASE II/III files, FW2 ☆ Mar:30
DELIMITED, DP □ Jan:64
file capture in emulation, FW2 § Jan:47
from a mainframe, DP □ Jan:64
Lotus 1-2-3 files into Javelin, UT ¥ Feb:59
SDF, DP □ Jan:64
SYLK files, FW2 ☆ Mar:38
WordStar, FW2 ☆ Mar:31
INDEX, DP § Jan:44, DP § Feb:48, DP ☆ Mar:9, DPA ☆ Mar:10, DP ☆ Mar:11, DP ○ Apr:58, DP ○ Jul:64, DP ○ Aug:67
Indexing, DP ¥ Jan:62
correctly updating with APPEND, DPA § Jul:22
file size, DP § Jan:44
numeric fields, DP ○ Apr:58
INKEY(), DP § Feb:48, DP ☆ Apr:16, DP ○ Apr:59, DP ○ Aug:67
Inkey.ASM, DP ◀ Apr:19,
See also: Leftarrow key
@inputline, FW2 ○ Mar:61, FW2 § Aug:53
INSTALL.BAT, DP § Aug:44
Installation, FW2 ☆ Jan:4, FW2 ☆ Jan:9
creating a shared dBASE Administrator directory, DPA ☆ Apr:11
creating shared database and program directories, DPA ☆ Apr:6, DPA ☆ Apr:12
custom drivers, FW2 ☆ Jan:16
dBASE Access, DPA ☆ Apr:7
dBASE Administrator, DPA ☆ Apr:6, DPA ☆ Apr:9, DPA ☆ Apr:11, DPA § Jun:27
dBASE III PLUS on the Starlan, DPA ☆ Apr:3
files required to run dBASE III PLUS, DP § Aug:44
@int(), FW2 ☆ Mar:26
INTENSITY= in CONFIG.DB, DP ☆ Jan:23

J

Javelin, UT ¥ Feb:59
JOIN
databases that contain memo fields, DP ☆ Aug:18

K

Key macros, RF ☆ Jun:45
Keyboard drivers, FW2 ☆ Jan:8
for Tandy 1200 HD, FW2 ☆ Mar:32
Keyboard LEDs, RF § Jul:53

L

LABEL FORM, DP ○ Jan:56, DP ☆ Jun:20
multiple labels, DP □ Jul:69
Labels, FW2 ☆ Mar:27
LASTDRIVE= in CONFIG.SYS, DP ☆ Jan:40
Layouts
saving, RF § Aug:61
Leading Edge Word Processor with dBASE III PLUS, DP ○ Jun:72
LEFT$(), RF § Jul:53

LEFT(), DP ☆ Jul:14
Leftarrow key
trapping it with Inkey.BIN, DP ☆ Apr:16
Library
macros, FW2 ☆ Mar:32
Library.FW2, FW2 ☆ Mar:33
Lines.ASM, DP ◀ Jun:15,
See also: Newsave.ASM
LIST
ALL, DP □ Feb:63
one page at a time with headings, DP □ Jul:70
TO PRINT, DP ☆ Mar:11
LIST PROCEDURE, DP § Apr:40
List.PRG, DP ◀ Jul:69
LOAD, DP ☆ Jan:28, DP ☆ Feb:21, DP ☆ Mar:18, DP ○ Apr:59, DP ☆ Jun:4
Locate, FW2 ☆ Mar:33
LOCATE, DP § Apr:41
Logical fields and variables, DP ¥ Jan:62
LOGIN.DB, DPA § Apr:34
Lotus 1-2-3, FW2 ☆ Feb:10,
See also: Range input
converting files from, DP ○ Aug:68
importing into Framework II, FW2 ○ Jun:73

M

Mac.INC, DP ◀ Jan:28,
See also: Screens
Macros, DP ☆ Jan:36, DP ○ Jan:58
append to cut paste buffer, FW2 § Jul:46
auto indentation, FW2 § Mar:55
convert spreadsheet cell to numeric constant, FW2 § Jul:47
database entry keyfilter, FW2 § Jul:44
dual column, FW2 ☆ Apr:21
key, RF ☆ Jun:45
remove blank lines from word frames, FW2 § Jul:45
scroll rate, FW2 § Apr:34, FW2 § Jun:76
wildcard directory information, FW2 § Jul:46
Mailmerge, FW2 ☆ Mar:27
from multiple databases, FW2 ☆ Jul:30
Make.PRG, DP ◀ Mar:53,
See also: CREATE
Makescr.PRG, DP ◀ Jun:16,
See also: Newsave.ASM
Makestru.PRG, DP ◀ Aug:27,
See also: Memojoin.PRG
MARGIN = in CONFIG.DB, DP ☆ Jan:23
Match criteria
using dates, RF § Jul:54
MAXMEM = in CONFIG.DB, DP § Feb:47
Mem2char.ASM, DP ◀ Aug:29,
See also: Memojoin.PRG
Memo fields, DP § Feb:48, DP ○ Apr:59
joining databases that contain, DP ☆ Aug:18
Memojoin.PRG, DP ◀ Aug:24
Memory, FW2 ☆ Jan:9
overlays, FW2 ☆ Jan:10
Memory addresses, DP ☆ Jul:4,
See also: DEBUG.COM
Memory variables
PUBLIC vs. PRIVATE, DP ○ Jul:63
MEMOWIDTH = in CONFIG.DB, DP ☆ Jan:23, DP § Feb:47
@menu, FW2 ○ Jan:58
MENU = in CONFIG.DB, DP ☆ Jan:23
Merge printing
multiple databases, FW2 ☆ Jul:30
Microsoft C Compiler
using with the Developer's Toolkit, FW2 ☆ Jun:30
MID$(), RF § Jul:53
@mid(), FW2 ☆ Mar:26
MOD(), DPA § Jul:25
MODE, FW2 ☆ Jan:8, DP § Feb:47, UT § Apr:32, UT § Apr:38
Modems
telecommunications questions and answers, FW2 ☆ Jul:36
MODIFY COMMAND, DP ☆ Jan:36, DP ○ Jan:57, DP ¥ Jan:62, DPA § Jul:25,
See also: Program editors
international characters, DP § Aug:43
MODIFY REPORT, DP ○ Jan:57
MODIFY SCREEN, DP ☆ Mar:10, DPA § Jul:25
MODIFY STRUCTURE, DPA ☆ Mar:10, DPA § Jul:25
MODIFY VIEW, DP § Jul:26
Month.FW2, FW2 ◀ Jun:38,
See also: Mymain.ASM
MontoNum.FW2, FW2 ○ Mar:61
Mouse, FW2 ○ Apr:60
MOVE, FW2 ☆ Jan:9
Multimerge.FW2, FW2 ◀ Jul:32
MVARSIZ = in CONFIG.DB, DP ☆ Jan:21
Mymain.ASM, FW2 ◀ Jun:32

N

Ndxsize.PRG, DP ◀ Jan:44
Negative numbers, RF § Jun:51
used as length arguments, RF § Jul:53
Newsave.ASM, DP ◀ Jun:6,
See also: Screens
@nextkey, FW2 ○ Jan:59
Norton Editor, DP ☆ Jan:37
Novell network, DP § Jan:46
MAP command, DPA § Aug:45
merge printing documents on, MM ○ Jun:75
SPOOL, DPA § Jul:20
Num2Word.ASM, DP ◀ Mar:19
Numeric coprocessor, FW2 ☆ Jan:8

O

ON ERROR, DPA § Jul:25
ON ESCAPE, DP ○ Mar:63, DP ○ Aug:66
.OR.
XOR, DP § Aug:40
Outline view, FW2 ☆ Mar:34
Overlays, FW2 ☆ Jan:10

P

@panel, FW2 ○ Mar:62, FW2 § Jun:43
PARAMETER
mispelling keyword, DP § Jul:24
Parameters
passing, DP ☆ Feb:22, DP ☆ Mar:18
PATH, UT § Apr:39
PATH = in CONFIG.DB, DP ☆ Jan:24
PEEK(), D2 ○ Apr:59
@performkeys, FW2 ○ Jan:58, FW2 ○ Mar:61, FW2 ☆ Jun:39
Point & dial, FW2 ☆ Jul:36
POKE, D2 ○ Apr:59
Port assignment, FW2 ☆ Jan:4
Print logic, MM ☆ Feb:3
PRINT = in CONFIG.DB, DP ☆ Jan:24
Printer
control, FW2 ☆ Jan:11
driver, FW2 ☆ Jan:4
Printer Action Tables, MM ☆ Apr:26, MM § Apr:42
Printing with dBASE III PLUS, DP § Apr:41
compressed mode, DP ○ Jul:64
field names as headers, DP ○ Jun:72
logical and character fields combined, DP ○ Aug:67
using European versions, DP ¥ Jan:61
@Z, DP ○ Jan:56
Printing with Framework II
a database in dBASE view, FW2 □ Jan:63
bolding frame labels, FW2 ○ Jun:74
driver for NEC 7710, FW2 ☆ Mar:35
dual column, FW2 ☆ Apr:21
elite type on the Okidata 193, DP □ Jul:71
footers, FW2 § Feb:50
format/control, FW2 ☆ Mar:35
frame labels, FW2 ☆ Mar:35
from virtual memory, FW2 ☆ Mar:35
headers, FW2 § Feb:50
maximum lines stranded, FW2 ○ Jun:73
orienting spreadsheets with SIDEWAYS, FW2 § Apr:34
page breaks, FW2 ○ Jul:66
removing /p, FW2 ○ Jun:73
row and column labels in a spreadsheet, FW2 ☆ Aug:47
selected records from a database, FW2 □ Mar:71
translating foreign languages, FW2 ☆ Jun:41
underlining headers, FW2 ○ Jun:74
using attributes, FW2 § Mar:56
Printing with MultiMate
creating endnotes, MM ☆ Mar:48
from dBASE files, MM § Jun:59
legal-size and letter-size documents, MM § Apr:42
math font cartridges, MM ☆ Apr:27
merging, MM § Mar:57
missing characters, MM § Apr:43
print attributes, MM ☆ Apr:26

print pitch table, MM ☆ Apr:28
Printer Action Tables, MM ☆ Feb:3, MM ○ Feb:58
using the HP LaserJet, MM ☆ Apr:26
using the WPS format, MM ☆ Mar:48
PROFS, FW2 ○ Jul:66
Program design
modularity, DP ☆ Feb:31
Program editors, DP ☆ Jan:35
WordStar, DP ○ Jan:57
Programming with
dBASE III PLUS
using European versions, DP ¥ Jan:62
PROMPT = in batch files, UT § Apr:32
PROMPT = in CONFIG.DB, DP ☆ Jan:21
PROTECT, DPA ☆ Mar:10
privileges, DP § Jan:45
Prtlab.PRG, DP ◀ Jul:69

R

RAM-resident programs
Sidekick!, DP ☆ Jan:38
Random Access Memory (RAM)
disk, FW2 ☆ Jan:10
Range input, FW2 ☆ Feb:10
Range.FW2, FW2 ◀ Feb:14,
See also: Range input
RangeInput.FW2, FW2 ◀ Feb:15,
See also: Range input
RapidFile
comparing dBASE III PLUS and RapidFile functions, RF ☆ Aug:54
READ, DP ○ Apr:59
Read-only files, RF § Jul:54
Referencing
library, FW2 ☆ Mar:36
regions, FW2 ☆ Mar:36
with English names, FW2 ☆ Mar:36
REINDEX, DP § Jan:44, DP ○ Jul:64
RELEASE ALL, DPA § Jul:25
REPLACE, DP § Feb:49, DPA ☆ Mar:10, DP ○ Jun:70, DP ○ Jun:71, DP □ Aug:72
ALL, FW2 ○ Mar:61
one character with another within a string, DP ☆ Jul:14
one string with another in a field, DP § Aug:42
Replace.ASM, DP ◀ Jul:17
Replace1.PRG, DP ◀ Jul:16,
See also: Replace.ASM
Replace2.PRG, DP ◀ Jul:17,
See also: Replace.ASM
REPORT FORM, DP ○ Jan:56, DP ○ Apr:59, DP □ Apr:64
floating dollar signs, DP § Aug:41
menu creation, DP ☆ Feb:30
retrieval specifications, DP ○ Aug:68
snaked column reporting, DP ☆ Jun:20
Report Statistics
handling using report fields, RF ☆ Jul:48
RESTORE.EXE, DP ☆ Jul:3
Restscr.ASM, DP ◀ Jan:32,
See also: Screens
RESUME, DP ○ Aug:67
RIGHT$(), RF § Jul:53
RIGHT(), DP ☆ Jul:14
Rptfrm.PRG, DP ◀ Feb:38,
See also: REPORT FORM
RUN, DP ☆ Jan:36, DP ¥ Jan:57
@run(), FW2 ☆ Mar:26

S

SAFETY = in CONFIG.DB, DP ☆ Jan:24
Sample.PRG, DP ☆ Aug:38,
See also: Getstat.PRG
Savescr.ASM, DP ◀ Jan:30,
See also: Screens
Saving
files, FW2 ☆ Mar:37
Sci.PRG, DP ◀ Mar:51,
See also: Scientific notation
Scientific notation, DP § Feb:49, DP § Mar:51
SCOREBOARD = in CONFIG.DB, DP ☆ Jan:24
Screens
AT&T PC 6300, FW2 ☆ Jan:18
avoiding snow on the CGA, DP ☆ Jun:6
color, FW2 ☆ Jan:9
driver, FW2 ☆ Jan:4
Hercules Graphics Card, FW2 ○ Jul:66
Hercules InColor Card, FW2 ○ Jul:66
high-resolution, FW2 ○ Jul:65
IBM Color Graphics, FW2 ☆ Jan:18
MDS Genius, FW2 ○ Jul:66
partial save and restore to files, DP ☆ Jun:3
saving and restoring to files, DP ☆ Jan:26
Sigma, FW2 ○ Jul:66
STB 400, FW2 ○ Jul:66
Wyse WY700, FW2 ○ Jul:65
Scroll rate macro, FW2 ◀ Apr:33
@sense, FW2 ☆ Jun:39, FW2 § Aug:52
Serial number protection, FW2 ☆ Jan:15
Serial port, DP ○ Jul:65
Serial.PRG, DP ◀ Jun:71
SET
ALTERNATE, DP § Jan:45
CARRY, DP ☆ Mar:13
CATALOG, DP □ Jan:64, DP § Mar:52, DP § Jun:28, DP □ Jul:68, DP § Aug:44
COLOR, DP § Jun:28, DP § Jul:28
DATE, DP ☆ Jan:24
DELIMITERS, DP ☆ Jan:21
DEVICE TO PRINT, DP ☆ Mar:12
DOHISTORY, DP ☆ Jan:24
ENCRYPT, DPA § Jul:20
ESCAPE, DP ☆ Apr:16
EXCLUSIVE, DP ☆ Mar:9, DPA ☆ Mar:10, DPA § Jul:20
FIELDS, DP ☆ Jan:24
FILTER, DP ☆ Jan:24, DP § Jun:29, DP ○ Jul:63
FIXED, DP ☆ Jan:24
FORMAT, DP ☆ Jan:24
HISTORY, DP □ Apr:64
INDEX, DP ☆ Jan:24, DP § Jun:28
MEMOWIDTH, DP ○ Apr:59
MESSAGE, DP ☆ Jan:24
ODOMETER, DP ☆ Jan:24
ORDER, DP ☆ Jan:24
PATH, DP ○ Jan:57, DP § Apr:39, DP § Jul:25
PRINT, DP ☆ Jan:24, DP ☆ Mar:12
PRINTER, DP ☆ Jan:24, DP ☆ Mar:12, DPA § Jul:22
PROCEDURE, DP ☆ Jan:24, DP § Apr:42, DP ☆ Jun:22
RELATION, DP ☆ Jan:24, DP □ Aug:72
saving and restoring environment, DP ☆ Aug:35
SCOREBOARD, DP ○ Aug:69
STATUS, DP ○ Jan:58, DP ○ Aug:69
TITLE, DP ☆ Jan:24
UNIQUE, DP ☆ Mar:11
@setselection, FW2 ☆ Feb:13, FW2 ☆ Jun:41, FW2 § Jun:42
Setup
changing configurations, FW2 ☆ Jan:8
for the first time, FW2 ☆ Jan:4
library of configurations, FW2 ☆ Jan:18
primary hardware, FW2 ☆ Jan:8
secondary hardware, FW2 ☆ Jan:8
SHELL = in CONFIG.SYS, UT § Apr:40
Sidekick!, DP ☆ Jan:38
SIDEWAYS, FW2 § Apr:34
SIZE, FW2 □ Jan:63, FW2 ☆ Mar:37
@Sizeto, FW2 ◀ Jun:43
Snake.PRG, DP ◀ Jun:23
Snapshots
saving, RF § Aug:61
SORT, DP ☆ Mar:10, DP ☆ Mar:12
Source, The
downloading articles from, ○ Aug:3
Spelling check
macros, FW2 ☆ Mar:28
personal dictionary, FW2 ☆ Mar:28
repeated words, FW2 ☆ Mar:28
SPOOL
using on the Novell Advanced NetWare network, DPA § Jul:20
using to print to a local printer, DPA § Jul:22
Spooling, FW2 ☆ Jan:10
SPORT.BIN, DP ○ Jul:65
Starlan network
adding users, DPA ☆ Apr:12
file server requirements, DPA ☆ Apr:4
subtracting users, DPA ☆ Apr:13
workstation requirements, DPA ☆ Apr:5
STATUS = in CONFIG.DB, DP ☆ Jan:24
STEP = in CONFIG.DB, DP ☆ Jan:24
STORE, DP § Jun:29
STR(), DP ○ Apr:58, DP ○ Aug:67
STUFF(), DP ☆ Jul:14
Subdir.PRG, DP ◀ Apr:36
Subdirectories in ASSIST, DP § Apr:35
SUBST, DP ☆ Jan:40

SUBSTR(), DP □ Feb:64, DP ☆ Jul:14
Substring search, DP □ Feb:63
Subtracting users on a network, DPA ☆ Apr:13
@sum, FW2 ☆ Feb:10, FW2 ○ Aug:69
@sumdate, FW2 § Apr:31
SUSPEND, DP ○ Aug:66, DP ○ Aug:67
Sylkexp.FW2, FW2 ◀ Mar:45, *See also: Exporting*
Sylkimp.FW2, FW2 ◀ Mar:41, *See also: Importing*

T

@t, FW2 ○ Apr:60, FW2 ☆ Jul:39
TALK = in CONFIG.DB, DP ☆ Jan:24
@tan(), FW2 ☆ Mar:26
Tandy 1000
  keyboard driver, FW2 § Jul:42
.TCM files, FW2 ☆ Mar:25
TEDIT = in CONFIG.DB, DP ☆ Jan:22, DP ○ Jan:57
Telecomm.TCM, FW2 ☆ Jul:38
Telecommunications
  auto line feeds, FW2 § Jan:46
  changing default settings, FW2 ☆ Jul:38
  questions and answers, FW2 ☆ Jul:36
  @t, FW2 ○ Apr:60
Terminal Emulation, FW2 ¥ Mar:64
Terminal emulation, FW2 ☆ Jul:41
  VT100, FW2 ○ Jul:66
@textselection, FW2 ☆ Apr:22
TimeFrame, FW2 □ Jan:63
@today, FW2 § Apr:31
Toshiba 1100 +
  Dnlevel/Uplevel, FW2 □ Feb:65
@trace(), FW2 ☆ Mar:26
TRANSFORM(), DP ○ Jan:56
Translation, foreign language
  FRED programs, FW2 ☆ Jun:39
TRIM, DP § Aug:41
TYPE(), DP § Jul:26
TYPEAHEAD = in CONFIG.DB, DP ☆ Jan:24

U

Underlining
  removing in long documents, MM § Jun:58
Uninstall, FW2 ☆ Jan:9, DP § Jun:29
  dBASE Access, DPA ☆ Apr:9
  dBASE Administrator, DPA ☆ Apr:15
UNIQUE = in CONFIG.DB, DP ☆ Jan:24
Upgrading
  from version 1.0, FW2 ☆ Jan:5

V

@var, FW2 ☆ Feb:10
VIEW, FW2 □ Jan:63
VIEW = in CONFIG.DB, DP ☆ Jan:24

W

WHILE, DP § Apr:41
@while, FW2 ☆ Jan:15
Word wrapping, DP ☆ Feb:21
WordStar, DP ☆ Jan:39
Wordwrap.ASM, DP ◀ Feb:24, *See also: Word wrapping*
Wordwrap.BIN, DP ◀ Feb:21, *See also: Word wrapping*
WP = in CONFIG.DB, DP ☆ Jan:22
Wraptest.PRG, DP ◀ Feb:27, *See also: Word wrapping*
@writetextfile, DP ☆ Mar:26

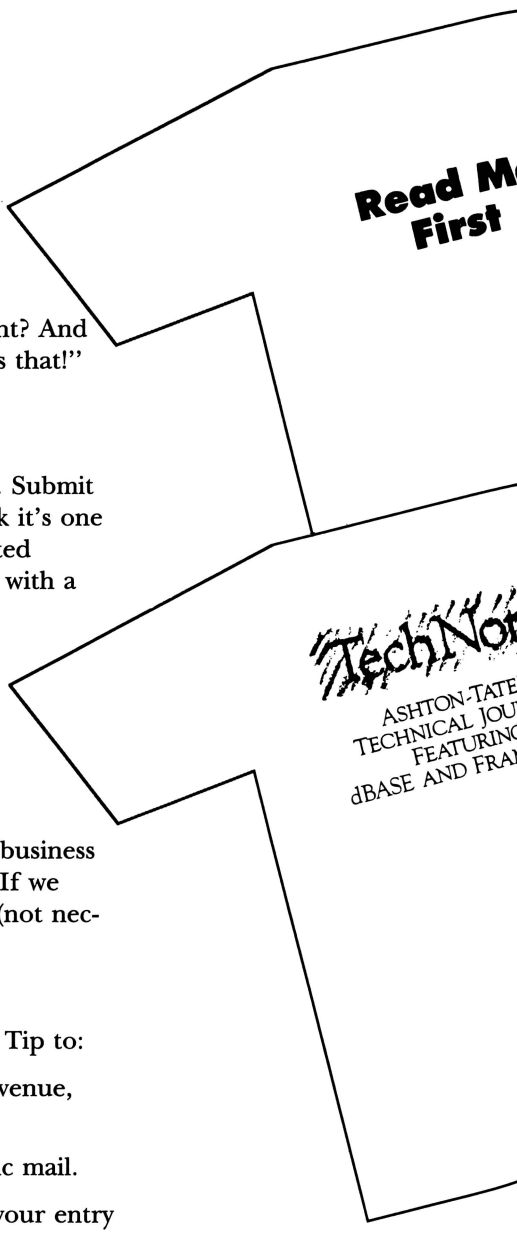
Y

Year.FW2, FW2 ◀ Jun:38, *See also: Mymain.ASM*

Z

ZAP, DP § Mar:52
Zapcis
  with CompuServe, FW2 ○ Jun:69
Zero
  dividing by, RF § Jun:50
  to the zero power, RF § Jun:51
ZOOM, FW2 ☆ Jan:9, FW2 ☆ Jan:15, FW2 ☆ Mar:37

We've Got Your F4 (Size)!



You've been a loyal reader of ***TechNotes*** for at least a couple of issues, right? And every time we print a Usage Tip you think, "Heck, I've got one as good as that!" Well, now you can prove it.

You (Almost) Can't Lose!

Leave it to ***TechNotes*** to dream up a contest where almost everybody wins. Submit *your* favorite Usage Tip for dBASE III PLUS or Framework II. If we think it's one of the ten best that we receive that month, we'll send you the highly coveted ***TechNotes*** t-shirt in the color scheme of your choice—as long as it's white with a blue ***TechNotes*** logo.

How will we know that your Usage Tip is one of the ten best? Easy:

It's going to be quick...
It's going to be handy...
It's going to be neat...
And most of all, it's going to be SHORT!

That's right, leave those major applications programs for your consulting business and send us the short, juicy stuff. Our judges have short attention spans. If we think it's good, we'll send you a fashion statement that will last a lifetime (not necessarily yours, though).

Here's how to enter:

Fill out the form below (or a facsimile) and send it along with your Usage Tip to:

TechNotes Tips Contest, Attn: Ken Getz, Ashton-Tate, 20101 Hamilton Avenue, Torrance, CA 90502-1319.

Or, if you're an on-line user, you can send your entry to us over electronic mail.

On CompuServe, type GO EASY at any FUNCTION: prompt, and leave your entry via EasyPlex for Ken Getz, 76703,3033.

On The Source, choose SourceMail from the opening menu, and address your entry to SIG007.

I warrant that by submitting a tip or tips to the TechNotes Usage Tips Contest sponsored by Ashton-Tate Corporation, that I own all rights to any entry submitted or I have the full written consent of any owners to submit the entry, and I agree to hold Ashton-Tate harmless from any claims due to submission of entries. I understand that Ashton-Tate may publish or otherwise disseminate information about the winning entries.

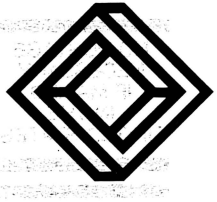
Signature _____ Date _____

Name _____

Address _____

City _____ State _____ Zip _____

Daytime Phone Number _____ T-Shirt Size_____



SUBSCRIBE TODAY TO TechNotes AND SAVE $22 OFF THE NEWSSTAND PRICE!

Get Ashton-Tate's hottest source of technical know-how at 30% off the newsstand price! Each monthly issue of **TechNotes** is filled with the most current, accurate technical information available on all dBASE, Framework, and RapidFile products. Subscribe today! Return this card today or call our toll-free numbers: **(800) 331-4164** (in California) or **(800) 828-2514** (residents of other states). In Canada, call **(619) 487-3212**. Newsstand price is $6 per copy (in U.S.A.).

☐ Bill me:
Amount enclosed:
☐ $50 (U.S.)
☐ $65 (Canada & Mexico)
☐ $75 (Other countries)
☐ New subscription
☐ Renewal

Form of Payment enclosed:
☐ Check (Make payable to Ashton-Tate in U.S. dollars drawn on a U.S. bank)
☐ Visa ☐ MasterCard
Expiration date:

Signature:

Name

Address

City State/Province

Zip/Postal Code Country

Phone Number

Which product do you use?
☐ dBASE II (D2) ☐ dBASE III (D3) ☐ dBASE III PLUS (DP)
☐ Framework (FW) ☐ RapidFile (RF)

Please allow 30 to 60 days for delivery of your first issue. 1000

READER SURVEY

We would like to make sure that ***TechNotes*** continues to serve the needs of our readership. Please take the time to complete this survey.

Which Ashton-Tate product do you use?
☐ dBASE II ☐ dBASE III/dBASE III PLUS ☐ dBASE III PLUS Lan Pack
☐ Framework II ☐ RapidFile

How long have you been using these product(s)? _____ years _____ months

How long have you been a subscriber to ***TechNotes***? _____

Rank in order of most usefulness (1-6):

_____ **Articles** _____ **Letters** _____ **Usage Tips**
_____ **Anomalies** _____ **On-Line** _____ **Corrigenda**

This article was/was not useful (because):

Fine-tuning the IBM PC Network Program _____

Joining Databases that Include Memo Fields _____

Generating SET Environment Programs _____

Printing Row and Column Labels in a Spreadsheet _____

Comparing RapidFile and dBASE Functions _____

IF YOU'RE MOVING

Don't forget to tell us so that we can be sure your subscription to **TechNotes** and/or the Ashton-Tate Quarterly is uninterrupted. To process a change of address, we must have your previous mailing address (including the zip code) and your new address.

My new address is:
Name/Department
Company Name/Add'l Destination Information
Mailing Address
City/State Zip Country

Please affix your current mailing label(s) here. (If the label is not available, be sure to include your old address.)

Affix your current Ashton-Tate Quarterly mailing label here

Affix your current TechNotes mailing label here

To change your address on the Ashton-Tate Product Mailing List, necessary to receive upgrade and other important information about your Ashton-Tate product(s), you must send a separate change of address to: Ashton-Tate Customer Service, 20101 Hamilton Ave., Torrance, CA 90502-1319.

SOFTSTRIP ™

We'
You

You've been a
every time we
Well, now you

You (Almost)

Leave it to *Tec*
your favorite U
of the ten best
TechNotes t-shi
blue ***TechNotes***

How will we kn

It's going to b
It's going to b
It's going to b
And most of a

That's right, le
and send us th
think it's good,
essarily yours,

Here's how t

Fill out the for

TechNotes Tips
Torrance, CA 9

Or, if you're a

On CompuSer
via EasyPlex fo

On The Source
entry to SIG00

- - - - - - - - - - -

I warrant that by sub
that I own all rights t
agree to hold Ashton-
publish or otherwise

Signature _____

Name _____

Address _____

City _____

Daytime Phone

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 66 TORRANCE, CA
POSTAGE WILL BE PAID BY ADDRESSEE

TechNotes

Subscription Service
P.O. Box 3729
Escondido, CA 92025-0929

Place
Postage
Here

TechNotes

Reader Survey
Ashton-Tate Software Support Center
20101 Hamilton Avenue
Torrance, CA 90502-1319

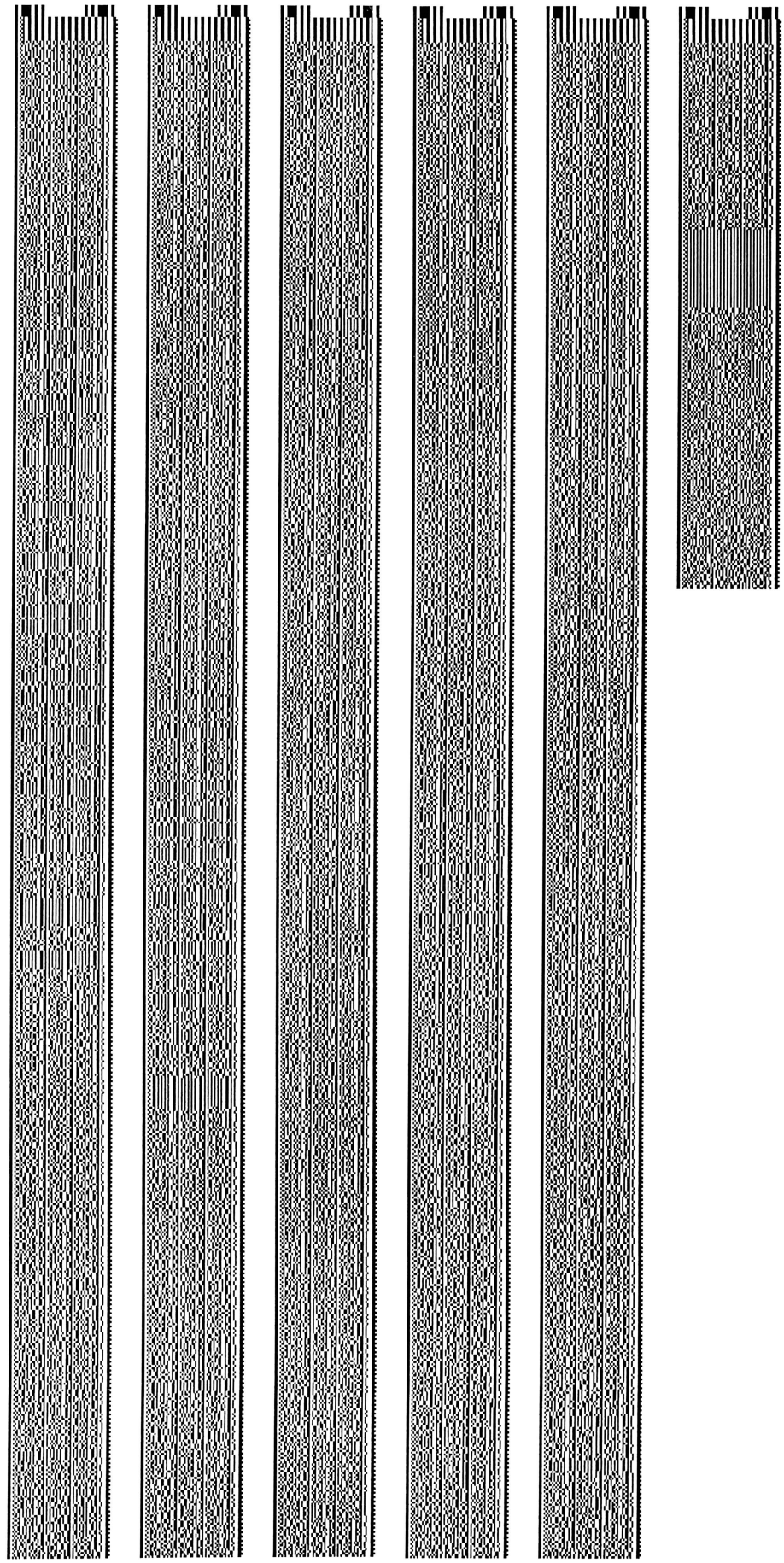
Attention: Brenda Johnson-Grau

Place
Postage
Here

Ashton-Tate Periodicals
PO Box 3729
Escondido, CA 92025-0929

ASHTON·TATE

FILES:
GETSTAT.PRG
CHKSTAT.PRG
SAMPLE.PRG
MEMOJOIN.PRG
CHKFILES.PRG
MAKESTRU.PRG
CHGTYPE.PRG
MEM2CHAR.BIN



1 2 3 4 5 6

SOFTSTRIP ™

TechNotes
P.O. Box 3729
Escondido, CA 92025-0929

Address Correction Requested

BULK RATE
U.S. POSTAGE
PAID
PERMIT NO. 324
GARDENA, CA